Technical User Guide

Customer Insight

Version 2.1

January 2021

Sabre Airline Solutions

Revision History

Revision Number	Date	Description
1.0	**September 2013**	• **Initial version**
2.0	**January 2019**	• **Major update across all content** • **Final review on 11th Feb 2019**
2.1	**January 2021**	**Update to ProfileIdentityInfoRead and ProfileDuplicateCheck services.**
2.2	**February 2021**	**Links and schema RQ/RS validation**

Table of Contents

Introduction 1

1.1 About this guide

This guide is for architects and developers to learn how to compose XML formatted requests and responses to consume Sabre Customer Insight Profiles Web Services.

The information in this guide is intended to help architects and developers accomplish the following:

- Incorporate into a client program the composition of the XML request and response formats to acquire appropriate Sabre Integrated Computing Environment (ICE) authentication and authorization security credentials represented by a unique token found in the response.
- Incorporate into a client program the composition of the XML to include the unique tokenized ICE security credential in subsequent Sabre Customer Insight Profiles Web Service requests.
- Consume Sabre Customer Insight Profiles Web Services by learning the different request types, their corresponding responses and the different protocols and standards involved.

Note This guide focuses on functions and elements that are intended exclusively for the use of *Sabre Airline Solutions* clients.

1.2 Standards and Specifications

The following specifications govern the format of profile data, the packaging format of the messages, and the transport mechanisms.

- HTTP 1.1 [RFC2616] – used for transport protocol
- HTTPS / TLS 1.2 (more information can be found at https://developer.sabre.com/docs/read/security/pci_mandate)
- MIME specifications [RFC2045], [RFC2046], and [RFC2387] – used for the SOAP message headers and instructions
- SOAP 1.1, Electronic Business using XML [ebXML], and W3C XML standards – used to define and describe the SOAP messages
- SOAP Messages with Attachments specification [SOAPAttach] – used for the ebXML messages that include header and payload containers
- WS-Security standards – partially adopted for some security elements in the SOAP header
- W3C XML 1.0 – used for both Sabre XML message specifications and Sabre XML messages put into the payload container of the SOAP message.
- Open Travel Alliance specification (http://www.opentravel.org) - used as the basis for the profile request and response XML payloads.
- SOAP – Simple Object Access Protocol
- Character Specifications
- ISO 10646/Unicode, Basic Latin and Latin 1 Supplement
- UCS Transformation Format 8 (UTF-8 Encoding)
- ISO 8859 Latin 1 Character Set

- Sabre Character Set

1.3 About Sabre *Customer Insight* Profiles Web Services (CI)

Web clients access the content and functionality of the *CI* web application by accessing the Web Services exposed through the Sabre Web Services (SWS) infrastructure.

The Sabre Web Services infrastructure provides security, sessions, logging, and routing to the *CI* web business services. A Web Services session is created when a correctly formatted SessionCreateRQ request is sent to the Sabre Web Services infrastructure, and a binary security token is returned in the SessionCreateRS message. This conversation ID and binary security token identify the session and are used together throughout the duration of the session.

In a given Web Services session, all request messages include the same unique conversation ID and binary security token. Only one conversation ID should exist per business process workflow. Once again, often the URL of the web client website plus a timestamp is used for the conversation ID.

If no activity occurred within the session time out limit, which is approximately 15 minutes, the business process flow represented by the session is not guaranteed to be alive.

A *simplified* flow of a Web Services session is as follows:

1. The web client first requests access, a connection, and a session by sending a SessionCreateRQ XML request to the Sabre Web Services infrastructure which does the following:

a. Validates the user security credentials and message format

b. Authenticates the message

c. Authorizes access

d. Preserves the requestor-composed unique conversation ID

i. Security credentials in the SessionCreateRQ message consist of the wsse:Username, wsse:Password, Organization, and Domain elements. In addition to these credentials, the client generates a unique conversation ID. Often the conversation ID achieves uniqueness by including a timestamp and the URL of the client website as part of the conversation ID.

ii. This is the only request message that includes the client security credentials provided by Sabre.The web client stores the session data represented as a conversation ID and security token, and includes them with each subsequent SOAP request until the conversation is closed. This avoids the overhead of re-authentication that would occur if this process needed to be launched for every type of *CI* service request.

2. The SWS infrastructure uses the security token and conversation ID to locate the Web Services session and if granted access, routes each of the subsequent Service requests to the appropriate *CI* Web Service and returns the response to the Web Client.
3. The Web Client and the *CI* services exchange messages that represent a business workflow.The web client sends subsequent *CI* Web Service requests incorporating the returned SWS security token as well as the unique conversation ID sent with the initial SessionCreateRQ request.
4. The Web Services session ends when the web client sends the SessionCloseRQ XML request with the same unique conversation ID and security token of the session it is closing, or the session time out expires before the next *CI* Service web request is received.

The web client delivers requests to the *CI* Web Services using the HTTPS protocol.

All requests are sent to a URL that is the single access point to Sabre Web Services.

1.4 Security

Sabre Web Services have implemented multiple layers of security for client applications. These layers include line security, authentication, authorization and confidentiality.

Line Security

Line security is the layer that secures the data traveling on the line over the Internet between external systems Sabre data centers and responds back to the external systems. *CI* Web Services support point-to-point synchronous transport HTTPS using TLS 1.2 with 256-bit encryption.

Clients who consume Sabre Web Services must implement line security with a secure sockets layer and they must secure the envelopes and payloads with HTTPS.

Authentication

Authentication is the layer that allows web client access to the *CI* Web Services. Security credentials are found in the wsse:Username, wsse:Password, Organization, and Domain elements present in the SOAP envelope in the SessionCreateRQ request message. The user receives the values for the security credentials when set up by Sabre to use the *CI* Web Services.

The Sabre Web Services infrastructure authenticates the service requestor using the security credentials found in the envelope of the request.

An XML fragment of the security credentials from the SOAP SessionCreateRQ message envelope is shown below:

```
<wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/12/secext"
xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/12/utility">
   <wsse:UsernameToken>
      <wsse:Username>USERNAME</wsse:Username>
      <wsse:Password>PASSWORD</wsse:Password>
      <Organization>USERORGANIZATION</Organization>
      <Domain>USERDOMAIN</Domain>
   </wsse:UsernameToken>
</wsse:Security>
```

Note The following is the second multipart MIME part containing the Sabre XML payload message:

```
<SessionCreateRQ>
   <POS>
      <Source PseudoCityCode="XX"/>
   </POS>
</SessionCreateRQ>
```

Authorization

The authorization layer gives web clients access to specific web services. When a web client sends a request, the Sabre Web Services infrastructure authorizes access to all services in the product packages to which an organization has subscribed. Currently, all *CI* Services are authorized as a package.

Confidentiality

The confidentiality layer maintains the privacy of the data in a payload during its transmission. *CI* Web Services use HTTPS with 256-bit encryption.

1.5 Network Connectivity

Access to the *CI* Web Services for web client applications is available through the Internet. Consequently, resources used to develop and deploy production applications must have Internet access.

If the system is behind a corporate firewall, information about the user's proxy server is required so that the applications can access the Internet.

1.6 Web Services Sessions

Sabre Web Services information in Sabre Dev Studio

For the most complete and up to date information on the use of Sabre Web Services, please consult Sabre Dev Studio at https://developer.sabre.com/.

If you do not have an account at Sabre Dev Studio please contact your *Sabre* Account representative or the technical support listed below in this document's introduction section.

Getting Started with Sabre APIs is available at
https://developer.sabre.com/resources/getting_started_with_sabre_apis/

Customer Insight resources and web services list can be found at:

- Resources https://developer.sabre.com/node/8717
- (you need to register first and log in)
- Web services list https://developer.sabre.com/ASProfilesProductSuite/references

Sabre Web Services Credentials

Sabre web services credentials for Airline Solutions customers normally are set under EPRs (Employee Profile Records) set up by the Airline customer in their corresponding airline partition. Once the airline client provides Sabre web-services activations / provisioning team the information on the given EPRs, these Sabre teams grant the required attributes to enable access to the required web services.

Therefore for the examples above for creating web service sessions, the following conventions apply:

- **Username** = is typically the Employee Profile Record (EPR) ID given for a user/client
- **Password** = Access password set for the EPR

- **Organization** =Organization / Pseudo-city assigned for the EPR (normally the same as the 2 letter airline IATA code and Domain
- **Domain** = normally the same as the 2 letter airline IATA code

Please contact your Sabre account representative to process the request.

SessionCreateRQ Request XML Example

The web client establishes a SWS session by sending security credentials and a unique conversation ID as shown in the following example.

The Envelope

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:eb="http://www.ebxml.org/namespaces/messageHeader"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsd="http://www.w3.org/1999/XMLSchema">
    <SOAP-ENV:Header>
        <eb:MessageHeader SOAP-ENV:mustUnderstand="1" eb:version="1.0">
            <eb:From>
                <eb:PartyId type="urn:x12.org:IO5:01">999999</eb:PartyId>
            </eb:From>
            <eb:To>
                <eb:PartyId type="urn:x12.org:IO5:01">123123</eb:PartyId>
            </eb:To>
            <eb:ConversationId>2012-12-15T11:15:12@clientURL.com</eb:ConversationId>
            <eb:CPAId>Sabre</eb:CPAId>
            <eb:Service eb:type="sabreXML">Session</eb:Service>
            <eb:Action>SessionCreateRQ</eb:Action>
            <eb:MessageData>
                <eb:MessageId>99999999</eb:MessageId>
                <eb:Timestamp>2012-12-15T11:15:12Z</eb:Timestamp>
                <eb:TimeToLive>2012-12-15T11:15:12Z</eb:TimeToLive>
            </eb:MessageData>
        </eb:MessageHeader>
        <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/12/secext"
xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/12/utility">
            <wsse:UsernameToken>
                <wsse:Username>EPR/UserName</wsse:Username>
                <wsse:Password>EPR/UserPassword</wsse:Password>
                <Organization>SabreSuppliedOrganization</Organization>
                <Domain>SabreSuppliedDomain</Domain>
            </wsse:UsernameToken>
        </wsse:Security>
    </SOAP-ENV:Header>
    <eb:Manifest SOAP-ENV:mustUnderstand="1" eb:version="1.0">
        <eb:Reference xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="cid:rootelement"
xlink:type="simple"/>
    </eb:Manifest>
    <SOAP-ENV:Body>
```

```
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The SOAP Payload

```
<SessionCreateRQ>
    <POS>
        <Source PseudoCityCode="SabreSuppliedPseudoCity"/>
    </POS>
</SessionCreateRQ>
```

A typical response is shown below.

```
<SessionCreateRS xmlns="http://www.opentravel.org/OTA/2002/11" version="1"
status="Approved">
    <ConversationId>2012-02-15T11:15:12@clientURL.com</ConversationId>
</SessionCreateRS
```

The SOAP Envelope

```
<soap-env:Envelope xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/">
    <soap-env:Header>
        <eb:MessageHeader xmlns:eb="http://www.ebxml.org/namespaces/messageHeader"
eb:version="1.0" soap-env:mustUnderstand="1">
            <eb:From>
                <eb:PartyId eb:type="URI">123123</eb:PartyId>
            </eb:From>
            <eb:To>
                <eb:PartyId eb:type="URI">999999</eb:PartyId>
            </eb:To>
            <eb:CPAId> SabreSuppliedPseudoCity</eb:CPAId>
            <eb:ConversationId>2012-12-15T11:15:12@clientURL.com</eb:ConversationId>
            <eb:Service eb:type="sabreXML">Session</eb:Service>
            <eb:Action>SessionCreateRS</eb:Action>
            <eb:MessageData>
                <eb:MessageId>97d9da35-3a36-4092-a934-99ba554ac712@73</eb:MessageId>
                <eb:Timestamp>2006-12-28T18:34:16</eb:Timestamp>
                <eb:RefToMessageId>99999999</eb:RefToMessageId>
            </eb:MessageData>
        </eb:MessageHeader>
        <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/12/secext">
            <wsse:BinarySecurityToken valueType="String"
EncodingType="wsse:Base64Binary">Shared/IDL:IceSess\/SessMgr:1\.0.IDL/Common/!ICESMS\/S
TSB!ICESMSLB\/STS.LB!-4617338326250370976!113241!0</wsse:BinarySecurityToken>
        </wsse:Security>
    </soap-env:Header>
    <soap-env:Body>
        <eb:Manifest xmlns:eb="http://www.ebxml.org/namespaces/messageHeader"
eb:id="Manifest" eb:version="1.0">
            <eb:Reference eb:id="SessionCreateRS" xmlns:xlink="http://www.w3.org/1999/xlink"
xlink:href="cid:SessionCreateRS">
```

```
            <eb:Description xml:lang="en-US">Session Create Response
Message</eb:Description>
          </eb:Reference>
        </eb:Manifest>
    </soap-env:Body>
</soap-env:Envelope>
```

Ending the Session

Once the web client establishes a SWS session, one or more profile requests can be made by the user. For example, an application might need to retrieve several customer profiles and update these profiles all in one session.

The SWS session ends in one of two ways: The user sends the **SessionCloseRQ** message populated with session data, or the SWS infrastructure ends the session when the session timeout has occurred.

Note If the web client user intends to send only a single profile service request, the SessionCreateRQ and SessionCloseRQ messages must still pack in the single request.

The **SessionCloseRQ** message for the example follows:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:eb="http://www.ebxml.org/namespaces/messageHeader"
            xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">

  <SOAP-ENV:Header>
    <eb:MessageHeader SOAP-ENV:mustUnderstand="1" eb:version="2.0">
      <eb:ConversationId>2012-02-15T11:15:12@clientURL.com </eb:ConversationId>
      <eb:From>
        <eb:PartyId type="urn:x12.org:IO5.01">clientURL</eb:PartyId>
      </eb:From>
      <eb:To>
        <eb:PartyId type="urn:x12.org:IO5.01">webservices.sabre.com</ eb:PartyId>
      </eb:To>
      <eb:CPAId>IPCC</eb:CPAId>
      <eb:Service eb:type="sabreXML">Session</eb:Service>
      <eb:Action>SessionCloseRQ</eb:Action>
      <eb:MessageData>
        <eb:MessageId>mid:20001209-133003-2333@clientURL</eb:MessageId>
        <eb:Timestamp>2012-02-15T11:15:12Z</eb:Timestamp>
        <eb:TimeToLive>2012-02-15T11:15:12Z</eb:TimeToLive>
      </eb:MessageData>
    </eb:MessageHeader>

    <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/12/secext"
              xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/12/utility">
      <wsse:BinarySecurityToken xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/12/utility"
                                wsu:Id="SabreSecurityToken" valueType="String"
                                EncodingType="wsse:Base64Binary">
Shared/IDL:IceSess\/SessMgr:1\.0.IDL/Common/!ICESMS\/STSB!ICESMSLB\/STS.LB!-
4617338326250370976!113241!0
      </ wsse:BinarySecurityToken>
```

```
    </wsse:Security>
 </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <eb:Manifest SOAP-ENV:mustUnderstand="1" eb:version="2.0">
      <eb:Reference xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="cid:SessionCloseRQ"
xlink:type="simple"/>
    </eb:Manifest>
  </SOAP-ENV:Body>

</SOAP-ENV:Envelope>
```

Note The following is the second multipart MIME part containing the Sabre XML message:

```
<SessionCreateRS xmlns="http://www.opentravel.org/OTA/2002/11" version="1"
status="Approved">
  <ConversationId>2012-02-15T11:15:12@clientURL.com </ConversationId>
</SessionCreateRS>
```

1.7 Credit Card Data Security

Sabre has implemented a number of measures to ensure Credit Card data security and compliance with PCI (Payment Card Industry) standards.

The following sections refer to measures that *CI* uses to ensure credit card data security at several levels.

Access

- All incoming and outgoing transmission of Credit Card (CC) data is through a secure channel. In Web Services, this is HTTPS; in batch files, this is SFTP + file encryption.
- SSH keys for SFTP are exchanged in a secure way and fixed for each transition channel.
- Control of IP address is implemented for each customer that tries to use *CI* Web Services with Credit Card data.
- Security keywords must be requested and approved for each customer who wants to read Credit Card data.
- Access requires a Sabre Host EPR CCVIEW keyword (Host/PSS access) or OSCCVIEW (Web Services only) security attribute which is used to access Sabre Web Services.
- Credit Card data is encrypted before it is stored in the database.

Storage and Internal Security Measures

- Encryption keys are stored on separate boxes. Access to obtain the keys is secured by different security certificates.
- Credit card data is never recorded in any log file. Only a “masked version” is available with the last four digits exposed.
- Sabre’s Development team does not have access to the database and application boxes.
- Detailed application logging tracks the user account for every access request for credit card data, even in encrypted form.

- The account used by the application to access the database controls access to enable only connections from *CI* Application boxes, thereby preventing access by any unauthorized source.
- The Password used by the application to access the database is stored in hashed form, so it cannot be read from configuration files and used by another application.
- Prior to any application release, system security audits are conducted to ensure continuous compliance.

1.8 Technical Support

There are several ways to obtain technical support. Please note that a Pseudo City Code or PCC is required. In case of Airline Solutions customer this code is normally the 2 letter Airline IATA code.

Note When reporting production or other critical issues, it is best to contact support by phone. Do not send an email.

Telephone

Note To ensure the most expedient response, you must submit all critical and high impact issues directly by phone to *Customer Care*.

(Information included in table below is also available in Release Notes documents and it was copied from a release note template version 4.9)

Call *Customer Care* at the following toll free number for your country:

Country	Toll Free Number
Antigua	888-832-4738
Argentina	0800-666-1664
Australia	1-800-081-993
Austria	800-291-705
Bahamas	1-800-389-0417
Bahrain	800-00-002 (WSC 5050)
Belarus	880-0114 PIN 375
Belgium	0800-77-029
Bolivia	800-10-0350
Brazil	0800-891-9210
Brunei	800-013 PIN 673
Canada	1-866-598-1706
Chile	800-412555
China	4001-202-315
Colombia	01-800-954-1326
Cyprus	800-96110
Czech Republic	800-700-117

Country	Toll Free Number
Denmark	808-85884
Egypt - Cairo	7955-770 PIN 5670
Egypt - Other	02-7955-770 PIN 5670
El Salvador	800-0000-0011
Estonia	800-12-122 PIN 5047
Finland	0800-914-860
France	0800-909-657
Germany	0800-181-7245
Greece	00800-16-122-055-533
Hong Kong	800-908-742
Iceland	800-8667
India	000-800-100-6116
Indonesia	001-803-016-1722
Ireland	1-800-657-198
Israel	1-809-246-033
Jamaica	1-866-402-6835
Japan	0053-116-0811
Korea	0030-813-1943
Malaysia	1-800-813-609
Malta	800-90112 PIN 356
Mexico	01-800-123-8537
Netherlands	0800-023-2237
New Zealand	0800-450-960
Norway	800-18-798
Pakistan	00800-9004-4226
Panama	00800-226-0662
Paraguay	009-800-598-1-0004
Peru	0800-52-226
Philippines	1-800-111-00338 or 1-800-111-00339
Poland	800-900-807
Russia	810-800-240-31012
Saudi Arabia	1-800-11 PIN 5671
Singapore	800-101-1651
South Africa	0800-980-981

Country	Toll Free Number
Spain	900-995-926
Sweden	0200-285-836
Switzerland	0800-894-354
Tahiti	888-832-4738
Thailand	1. Dial 1-800-000-133 (AT&T) 2. Wait for the recording asking for the number you are dialing. 3. Dial 888-832-4738.
Trinidad and Tobago	888-870-9002
UAE	800-035-702-569
UK	0800-0288446
Uruguay	2518-6642
USA	1-888-421-8889 or 1-800-677-0856
Venezuela	0800-100-3851
Vietnam	1. Dial 1-201-0288 (AT&T Toll Free Number). 2. Wait for the recording asking for the number you are dialing. 3. Dial 866-947-8059.
Countries with no-toll free service	+1 770 261 0080 (toll call).

You can also use the **Call Me** button when you need a *Customer Care* analyst to call you back. You can access the **Call Me** button from the following two locations on the *Community Portal*:

- On the **Home** page, in the **Support Services-eService tool** area.
- On the Contacts page, in the **Customer Care** area.

Or call your regional Sabre software help desk

Email

Send an email to the following address: webservices.support@sabre.com

Email is monitored Monday through Friday during extended business hours.

Additional Support

Sabre maintains a Product Support Desk that is staffed 24 hours a day, 7 days a week to receive and resolve user-reported problems with various Sabre products. If you have questions, please contact the Product Support Desk.

Email account: sabresonic.helpdesk@sabre.com

Accessing CI Web Services 2

2.1 Activation

To consume Customer Insight data, the user must have a Sabre Web Services account. This account will have user credentials and be assigned under a specific *Domain* (normally set after the 2 letter IATA airline code), as well as a Group also referenced as suffix or iPCC (internet Pseudo City Code) which is normally set for Airline Solutions customers after the 2 letter IATA airline code. Accounts can be either set up as TPF (host) user, in case they need to access Sabre host functionality like PNR and move profile information into a PNR, or a non-TPF account, which is set up in ICE (Sabre's Integrated Computing Environment) with access only to Open Systems functionality in Sabre Web Services.

Additionally the web service request needs to include a request to access *Customer Insight as External User*. Processing this request will result in adding the corresponding attributes that manage the permissions to these services. In order to perform operations in the *CI* system, it is also necessary that the Domain (airline code) is enabled in the *CI* system.

Please contact your Sabre account representative to process the request. Upon completion of this activation process, Sabre will send a confirmation email indicating that the Domain has been activated and is ready to consume *CI* web services.

The ICE attributes which are required are listed below:

- **USGSessionUser** (typically assigned to an individual user or Employee Profile Record / EPR)

  Required to be able to create webservices session

- **CustomerInsightExternalUser**

  Required to consume CI web services for ProfileCreate, ProfileRead, ProfileUpdate, ProfileDelete, ProfileSearch, etc.

In addition, the following ICE attributes might be requested and assigned to the ICE account:

- **OSCCVIEW** (assigned at the Domain level for all users or at the individual user account / EPR level)

Control a view rights for Credit Card data stored in CI.

- **BTFOPView** (assign at the Domain Level for all users or at the individual user account / EPR level)

  Control a view rights for Travel Bank / Agency Manager account number stored in CI.

If user has assigned BTFOPView Ice attribute than he will be able to view Travel Bank/Agency Manager account number and CC data will be returned masked.

If user has assigned an OSCCVIEW Ice attribute then he will be able to view both Credit Card number and Travel Bank / Agency Manager account number stored in CI.

2.2 Types of Web Services

A web client that consumes *CI* Web Services typically uses two basic types of messages: session management messages and *CI* service request messages.

Session Management Services

Web Services messages have been created to open and close sessions. Session services do not request profile-related content from *CI*, and they do not contain business logic. Session services allow for greater efficiency when processing client requests because time consuming security data retrieval overhead is avoided by maintaining that data in the session and representing the session in a security token returned to the client.

The **SessionCreateRQ** request contains a client composed unique **conversation ID** and a Sabre supplied **security credential** to initiate a connection, a session, authentication and authorization to access particular Sabre services such as the *CI* Web Services.

The **SessionCreateRS** response returns to the web client the unique **conversation ID** and a unique **security token** for use in subsequent *CI* related requests. The session lasts either until the client sends a **SessionCloseRQ** request or the session time out is exceeded.

Establishing a session for *CI* services includes granting access only to the "owner" of the profiles stored by Sabre. This limited access is accomplished with the security configuration represented by the security credentials presented by the web client in the **SessionCreateRQ**.

Customer Insight – Profile Services

CI web clients can request via the internet the following profile services:

- **Profile Create** – the web client sends profile data to Sabre to create a profile to be stored by Sabre on behalf of an airline. The client sends the **Sabre_OTA_ProfileCreateRQ** XML request message and gets the **Sabre_OTA_ProfileCreateRS** XML response message back indicating success or failure and some profile-related information.
- **Profile Update** – the web client sends updated profile data to Sabre to modify an existing profile. The client sends the **Sabre_OTA_ProfileUpdateRQ** XML request message and gets the **Sabre_OTA_ProfileUpdateRS** XML message response indicating success or failure and some profile ID related information.

  The **Sabre_OTA_ProfileUpdateRQ** XML request can be composed to perform a full data update.

  When a full profile data is successful the updated profile data provided replaces the profile stored in Sabre. The web client must read (retrieve) the profile prior to fully updating it. The retrieved profile contains a timestamp that must be inserted into the profile update message or the profile will be rejected. The returned timestamp provides a means to guarantee data integrity. If another web client has updated the profile prior to this client returning the update, an error response message "**Invalid Last Update Time Stamp. Profile probably has been updated by another Client**" is returned due to the timestamp mismatch. If this update error is returned, the client must retrieve the profile again, update the profile with the newer timestamp, and update the profile with the client updates.

The Sabre_OTA_ProfileUpdateRQ has ability to update only selected subject areas and do not overwrite those which are not included in xml update request. In order to do so user must include the subject area he wants to edit and those which must remain unchanged need to be listed under <IgnoreSubjectArea> element.

Many subject area names that can be used within IgnoreSubjectArea. Please refer to the schema files for a complete list of supported areas.

Additionally, Sabre has developed the ability to conduct "Partial Updates". This allows the user to specify the elements that need to be added/modified or removed in the payload, without having to pass the full profile data in the request. However, the entire data model is not yet supporting Partial Updates (see more details in the "Update Service" section further below in this document).

Profile Read – the web client sends a request to retrieve a profile using the **Sabre_OTA_ProfileReadRQ** XML request message.
Sabre_OTA_ProfileReadRS contains the latest version of the profile stored by Sabre.

The **Sabre_OTA_ProfileReadRQ** message contains a <TPA_Identity> element with attributes to specify the profile ID, the profile type, the "owner" code (DomainID), etc. Therefore the same message format is used to retrieve any of the profile types such as traveler, corporate, travel agency, travel agent or alliance profile.

Sabre_OTA_ProfileReadRQ also has the ability to read specified subject areas and ignore other subject areas. The <PartialReadSubjectAreas> element under <Profile> indicates which subject areas are of interest in the profile. If a profile has subject areas specified in <PartialReadSubjectAreas>, then the Profile read request returns only that subject area(s). It ignores all other subject areas in the profile. If a profile does not have corresponding subject area specified in <PartialReadSubjectAreas>, then the Profile read request returns <TPA_Identity> section.

<IgnoreReadSubjectAreas> tells what subject areas are not of interest in the profile read. In this case, the Profile read returns all subject areas except the subject areas mentioned in <IgnoreReadSubjectAreas>.

A list of subject area names that can be used within PartialRead and IgnoreRead is found below:

- CustomerValueScore
- RelatedIndividual
- CustLoyalty
- PersonName
- EnrollmentInfo
- PaymentForm
- Address
- Email
- Telephone
- Login

The **Sabre_OTA_ProfileReadRS** message contains all the profile data retrieved or an error message indicating a problem was encountered while reading the profile.

- **Profile Search** – the web client sends search criteria for a profile in the **Sabre_OTA_ProfileSearchRQ** XML request message. If there is only one profile that matches the search criteria, the entire profile is returned to the web client in the **Sabre_OTA_ProfileSearchRS** XML response message. Otherwise, if the ProfileNameOnly attribute (in the **Sabre_OTA_ProfileSearchRQ** XML) is set to "*true*", a list of *TPA_Identity* sections of profiles that match the criteria is returned. If this attribute is set to "*false*", the user will get a list of *TPA_Idenity* sections along with some additional profile data. The search criteria entered depends on the type of profile as does the response data returned.

- **Profile Delete** – the web client sends a request to mark an existing profile for deletion in the **Sabre_OTA_ProfileDeleteRQ** XML request message. When the profile is marked for deletion, it gets the Profile StatusCode set as "DL" and will be permanently removed from the database when the *Purge Service* is executed. The Purge Service starts automatically at midnight (US Central standard time) and removes all the "marked for deletion" profiles that either have the same purge date as the current date or were not set a specific purge date at all. To program a Delete request in the future, the user can set the value of "PurgeDays" to r the number of days when they want to perform the Delete request.
  Note: *Depending on the time of the Profile Delete action, the Purge process would execute approximately after 48hrs to remove the profiles from the database*.This service also supports the ability to *Restore* a profile that is marked for Delete (StatusCode="DL") prior to the time of the Purge execution.

- **Profile Merge Service** – This service a client that has already identified the existence of one or more duplicate profiles for the same individual, to merge the profiles by calling **Sabre_OTA_ProfileMergeRQ**. This means that the profile identified as "D*uplicate*" will be de-activated and the profile identified as "Master", will be retained active in the system and will include an association to the duplicate profile that was de-activated. Additionally this service will update the Profile Index for PNRs that had been indexed with the duplicate profile, replacing the index with the profile ID of the Master profile.

- **Profile Identity Info Service** – the web client sends a request to obtain passenger name and tier level information of the Frequent Flyer by calling **Sabre_OTA_ProfileIdentityInfoReadRQ**. Having specified Frequent Flyer number and DomainID (airline code) user can obtain passenger name and tier level indicator details stored in *CI* as well as tier level description i.e. Gold member, Silver member. Additionally, if configured for an airline client, this service also allows to obtain the "KnownTravelerNumber" and "KnownTravelerIndicator" for customers storing such information as part of the Transportation Security Administration expedited screening initiative.

  Customer is also able to obtain tier level information for their partner airlines. However, access to other airline information must be granted by the corresponding partner airline and once approved, would need to be set up by the Customer Insight Delivery Team.

  Please note that even though access is granted, the host airline is only able to obtain limited partner profiles information, which is: Passenger name and tier level.

- **Profile Duplicate Check Service** – can be used as a standalone service to verify if a profile duplicate or possible duplicates of a profile already exists in *CI* or if it should be created.

  In the Profile Duplicate Check Service XML request customer can specify selected profile elements compliant with **Sabre_OTA_DuplicateProfileCheck** schema.
  Duplicate check request is processed based on the duplicate check pattern selected by customer and set up in *CI* database.

2.3 Message Structure

Messages for the *CI* Web Services conform to the following two specifications:

- The ebXML part of the SOAP envelope conforms to SOAP with Attachments.
- The content of the payload attachments conforms to CI XML which is based on, but not 100% compatible, with the published OTA profile related schemas.

The structure of the message is based on Internet standards such as HTTP, HTTPS, and the MIME mail extensions. HTTPS is the communication protocol.

The SOAP with Attachments protocol is a MIME multipart message with the following MIME parts:

- The header container – This is a SOAP envelope, which is an XML document.
- The payload container – This is the application payload, and it is formatted as CI XML.

The SOAP with Attachments protocol is used to format messages for Java clients, and the payload is sent as an attachment. Instead of sending the payload as an attachment, however, it can be included inside the SOAP wrapper. Java Axis clients include the payload inside the SOAP wrapper. If WSDL and Microsoft .NET Framework are used to format messages, the payload is included inside the SOAP wrapper.

2.4 Requesting Payload Content

Payload content is requested by including the action code that corresponds to the service being called. Service Names will not match Action codes but will be similar in naming convention.

A unique action code identifies the request and response payloads for every one of the *CI* Web Services.

The action codes, represented by the **eb:Action** element in the SOAP header, are the following:

Service Name	Action code
Profile Create	EPS_AS_EXT_ProfileCreateRQ
Profile Read	EPS_AS_EXT_ProfileReadRQ
Profile Search	EPS_AS_EXT_ProfileSearchRQ
Profile Update	EPS_AS_EXT_ProfileUpdateRQ
Profile Delete	EPS_AS_EXT_ProfileDeleteRQ
Profile Merge	EPS_AS_EXT_ProfileMergeRQ
Profile Duplicate Check	EPS_AS_EXT_DuplicateCheckRQ
Profile Identity Info Read	EPS_AS_EXT_ProfileIdentityInfoReadRQ

2.5 Web Services Error Handling

Several error categories are possible:

- **Sabre Web Services errors** – These type of errors occur within the Sabre Web Services infrastructure, and they are caused by faulty messages from the web client or problems with the *CI* Web Services. The infrastructure detects and generates these errors, and returns them as SOAP faults, with or without ebXML headers.
- **Business application errors** – These errors are generated by the business application services that are called by the SWS infrastructure. The web client request, the network routing between the SWS infrastructure and the business application service, or the business application service can cause these types of errors. They are returned to clients in the ErrorRS XML response format.
- **System errors generated by web clients** – These are caused by the web clients themselves and are external to the *CI* Web Services. They normally occur in the development environment, and are returned to the client.

When the response contains the <soap-env:fault> element, an HTTP status code of 500 is returned. If no SOAP fault exists, HTTP Status Code 200 is returned. Other codes depend on the request and are shown later in the document.

Some common error messages are listed below, such as those returned by CreateSessionRQ when:

- wrong values are passed in Organization / Domain
- when the account has not been activated for SWS or
- when the passcode is locked or invalid

Example:

```
  <wsse:Security xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/12/secext"/>
   </soap-env:Header>
   <soap-env:Body>
     <soap-env:Fault>
       <faultcode>soap-env:Client.AuthenticationFailed</faultcode>
       <faultstring>Authentication failed</faultstring>
       <detail>
         <StackTrace>com.sabre.universalservices.base.security.AuthenticationException:
errors.authentication.USG_AUTHENTICATION_FAILED</StackTrace>
       </detail>
     </soap-env:Fault>
   </soap-env:Body>

</soap-env:Header>
   <soap-env:Body>
     <soap-env:Fault>
       <faultcode>soap-env:Client.InvalidSecurityToken</faultcode>
       <faultstring>Invalid or Expired binary security token: {0}</faultstring>
       <detail>
         <StackTrace>com.sabre.universalservices.base.security.AuthenticationException:
errors.session.USG_INVALID_SECURITY_TOKEN</StackTrace>
       </detail>
     </soap-env:Fault>
   </soap-env:Body>
```

The example below illustrates the error response when DomainID has not been set up as “Active” (status AC) and therefore is not allowed to perform transactions in the *CI* system.

```
<soap-env:Body>
  <Sabre_OTA_ProfileCreateRS
   TimeStamp="2010-03-16T15:40:27.142" Version="6.49"
    xmlns="http://www.sabre.com/eps/schemas">
   <ResponseMessage>
    <Errors>
     <ErrorMessage>C:::PRF::Domain Status is not allowed to do this operation.</ErrorMessage>
    </Errors>
   </ResponseMessage>
   <Profile
    ClientCode=“AS” ClientContextCode="EXT" DomainID="XX"
    ProfileNameModifyIndicator="Y" ProfileStatusCode="AC"
```

```
   ProfileTypeCode="TVL" UniqueID="*"/>
 </Sabre_OTA_ProfileCreateRS>
</soap-env:Body>
```

CI Profile Types 3

3.1 Profile Types

CI offers 5 profile types: Traveler, Corporate, Travel Agency, Travel Agent and Alliance. Each of these profile types is described in more detail in the following sections. Each Profile Type contains elements in the schema referred to as "subject areas." Some examples of elements defined as subject areas would be Telephone, Email, Address, CustomerReferenceInfo, and Discounts.

3.2 Traveler

CI clients use the ***Traveler*** (TVL) profile type to store data applicable to an individual person, and could contain information such as name, address, email address, phone number, customer loyalty programs, or preferences. The ***Traveler*** profile can contain data for one or more of these purposes.

Traveler profile information stored in *CI* is integrated with SabreSonic Interact and SabreSonic Web.

Note For current Profiles-supported elements, attributes, and number of instances allowed of these fields, please refer to the schema xml files located on the Sabre Dev Studio.

Logic for specific schema elements/attributes within the Profile

Note ***PersonName*** should be used to identify the single individual/entity.

Instances of *PersonName* can only be used in combination with Language selection.

Customer Insight is able to support up to 25 instances of PersonName which can be used in combination with Language selection with the intent to illustrate a different representation of the same person's name in different languages. If the LanguageIDCode is not specified, it will be set to the default code which is 'EN-US'

These features can be leveraged for customer integration with their own systems or 3rd party tools. However, since SSW and Interact applications only make use of single instance of name and do not offer full support for non GDS characters, if *CI* is used in combination with these and other CSS products / interfaces, it is recommended to use only GDS supported characters.

OrderSequenceNo Uniqueness based upon predefined Subject Area 'Types'

While the OrderSequenceNo attribute is optional in all locations in the Profile schemas, it is highly recommended that customers use this attribute in the service calls when creating and updating Profiles.

In most cases, subject areas in the schemas support the ability for the user to store multiple instances of data. When multiple instances are stored for the same subject area, there is a need to identify each instance with an **OrderSequenceNo**. This helps the user to ensure that when updates to the profile are made, the correct instance of that subject area is updated. It also helps point of sale tools with their display logic to group the data together for easier interpretation and execution for any needed action by the user.

Similarly, these subject areas supporting multiple instances where attribute "**OrderSequenceNo"** is available also have "**DisplaySequenceNo**" optional attribute. If using CI in combination with

SabreSonic Interact and SabreSonic Web it is recommended to populate both these attributes with the same numeric value 1 to n.

3.3 Corporate

CI clients use the ***Corporate*** (CRP) profile type to store and manage information about a corporation with whom the airline has a relationship or serves as a customer to provide business travel services. Traveler profiles can be associated to Corporate profiles to establish a relationship between the two.

Corporate profiles stored in *CI* are integrated with SabreSonic Web Corporate Storefront.

NOTE: For current Profiles supported elements, attributes, and number of instances allowed of these fields, please refer to the schema xml files located on the Sabre Dev Studio.

3.4 Agency

CI clients use the ***Agency*** (AGY) profile type to store and manage information about the Agency servicing the airline. This can include Agency data such as contact numbers, emails, addresses as well as vendor preferences and discount programs, etc.

Agency profiles stored in *CI* are utilized by SabreSonicWeb Agency portal.

Note For current Profiles supported elements, attributes, and number of instances allowed of these fields, please refer to the schema xml files located on Sabre Dev Studio.

3.5 Agent

CI clients use the ***Agent*** (AGT) profile type to manage data about travel agents authorized to handle travel through the travel agency. Typically the travel agent profile is related to a *CI* travel agency profile. The travel agency can be enrolled in an airline incentive program via SabreSonicWeb Agency portal.

Note For current Profiles supported elements, attributes, and number of instances allowed of these fields, please refer to the schema xml files located on Sabre Dev Studio.

3.6 Alliance

Alliance profile type (ALC) is used to store profile information for Sabre non hosted airlines. ALC profiles stores very limited customer information like passenger name, Frequent Flyer number and tier level information. These profiles are stored in *CI* in order to perform name/number validation and tier level recognition only when frequent flyer information is entered into a PNR. Example: if Sabre hosted airline "XX" is a member of Star Alliance then *CI* will store records of all Star Alliance partners' top tiers as Alliance profiles. If Agent during a booking will make a FF entry to add Frequent Flyer number into PNR then if a record exists in *CI* then a corresponding banner i.e. GOLD will be displayed and stored in PNR.

Airline / agent clients do not have a direct access to Alliance profiles since the use of these profiles is currently restricted for access only via PNR as indicated above.

3.7 Common attributes within the Payload

The following attributes are available for every Profile Type:

Attribute	Description
ClientCode	Indicates the code that is associated with the profile that is shared between calling applications.
ClientContextCode	It is the calling application code.
UniqueID	Profile ID.
ProfileTypeCode	Type of profile as described further.
ProfileName	Name of the profile if needed.
ProfileNameModifyIndicator	Indicates if profile name can be modified or not.
ProfileDescription	If necessary describes the profile.
DomainID	This will be carrier code for AS profiles.
ProfileStatusCode	Indicates the status of the profile as described further.
ProfilePurgeNoDays	Number of days after which the profile should be purged.

ClientContextCode

Each customer connecting *CI* will be provided with a dedicated ClientContextCode. This attribute acts as an identifier for any action made by the system, however it does not act as an “owner” of the object identified in the service call.

As an example, an airline may use profiles via SabreSonic Web, SabreSonic Interact and their own Loyalty System. If all of the applications will be sending service calls for any actions on profiles (Create, Update, Read etc.) each system will be provided with a dedicated ClientContextCode (CCC) that should be included in the payload as an identifier. An external Loyalty System can create the profile sending a ClientContextCode=”EXT”, SabreSonic Web is assigned CCC=”SSW” and Interact is assigned its own dedicated CCC=”INT”.

3.8 Profile SubTypes

A Profile**SubType** is an optional element under the **TPA_Identity** section of the Profile node that can be used by consuming applications.

These subtypes are simply used as additional qualifiers to designate certain qualities or functions for consuming applications and/or *CI* system in an individual’s profile including Notification Services. The profile consuming applications like SSW or Interact are using those values as qualifiers to implement business logic on their end.

Profile type and profile subtype are also used to configure profile batch extract files that can be provided by CI.

Even though profile subtype is optional attribute it is highly recommended to include them in the request payload.

The following subtypes are available:

For Type= TVL (Traveler)

1. "**FFP**" - Used for all frequent traveler profiles for the loyalty system belonging to owning airline including top tiers and base members.
2. "**WEB**" - Used normally for those profiles created by an Internet booking engine (i.e. SabreSonic Web "SSW") that are not Frequent Flyer profiles or another source when a passenger accepts to retain their customer contact data in a profile record for later use but that are not enrolling in the Loyalty program of the hosted airline.
    a. Business Traveler "**BTV**" - Used to identify a traveler that is associated to a Corporation. For example in SabreSonicWeb Corporate Loyalty, if passenger is considered as Business Traveler and is associated to a Corporation additional fare discounts might apply to his fare during a booking process.
        i. Typically created / managed by SabreSonic Web Corporate Loyalty and when used with this product this profile contains an "association" to a CRP profile.
        ii. BTV profiles can also be retrieved and edited by Interact. If the passenger is associated to Corporate profile then CorporateID is displayed in Interact profile page.

            Frequent Flyer profiles (FFP), WEB and Business Traveler (BTV) profiles can be utilized via Interact.
    b. Corporate Administrator –"**CAD**" – Is created for those customers who are authorized to manage Corporate profiles via SSW Internet Booking Engine (IBE) and designate others to act as Travel Arrangers.

Corporate Administrator profile is created automatically when Corporate profiles is created via SSW SAT (Site Administration Tool). Each Corporate profile managed via SSW may have only one Corporate Administrator profile associated to.

For Type= AGT (Agent)

Travel Arranger "**AGR**" ProfileSubTyp "subtype"– usually created for a person who is authorized to make travel arrangements for other travelers (typically for business travelers)

Typically created / managed by SabreSonicWeb Agency portal (Agency Manager)

3.9 How to Determine the Profile Data Needed

This document contains many examples of XML and additional examples are available on Sabre Dev Studio for different profile types and objects available within *CI*.

Most likely an Airline new to the *CI* system will have data requirements that will be a smaller subset of all the hundreds of possible data types. You will find that most sections in the different profile types are optional, and therefore each client can decide which sections and elements are needed for their particular business needs.

Here are some tips to make the use of this document a little easier:

- Go to the session management section and understand the composition and use of the **SessionCreateRQ** request and **SessionCreateRS** response.
- Go to the section of the document discussing Profile Types.

- Look at a typical subset of data for a particular profile type and determine if it is a good match for your business requirements.
- Look at the create sample XML for the profile type and review the additional data types available and include those data types, if desired.
- Compose create, update, retrieve, and search XML messages that represent the subset of data types chosen.
- In case you have any questions or issues, you can contact Sabre for support at webservices.support@sabre.com and review the set of composed XML messages.
- Setup Customer Acceptance Testing (CAT, a.k.a. Customer Validation Testing CVT) with Sabre.

3.10 Use of Dictionary Control Table Values

In order to maintain data standardization, *CI* has implemented a number of "dictionary tables" for controlled table values. These controlled table values prevent the user from entering incorrect or inconsistent values in the XML messages for profile data fields. There are a number of Control Data values that are used in the profiles services and these include, for example, the profile Type Codes (i.e. TVL for Traveler, CRP for Corporate, etc.) Credit Card Type codes to differentiate between credit cards, debit cards, etc., Airport and City codes, etc.

Examples of this type of data include:

CardTypeCode.xml
<CARDTYPECODE>CR</CARDTYPECODE>
<DESCRIPTION>Credit</DESCRIPTION>
or
<CARDTYPECODE>DB</CARDTYPECODE>
<DESCRIPTION>Debit</DESCRIPTION>
Etc.

ProfileTypeCode.xml
<PROFILETYPECODE>TVL</PROFILETYPECODE>
<DESCRIPTION>Traveler</DESCRIPTION>
or
<PROFILETYPECODE>CRP</PROFILETYPECODE>
<DESCRIPTION>Corporation</DESCRIPTION>
etc.

DocTypeCode.xml
<DESCRIPTION>Drivers License</DESCRIPTION>
<LANGUAGEID>EN</LANGUAGEID>
<DOCTYPECODE>DRLS</DOCTYPECODE>
</DOCTYPECODE>
or
<DESCRIPTION>National Identity Card</DESCRIPTION>
<LANGUAGEID>EN</LANGUAGEID>
<DOCTYPECODE>NTID</DOCTYPECODE>
</DOCTYPECODE> etc…

Address_LocationTypeCode.xml
<ADDRESS_LOCATIONTYPECODE>
<DESCRIPTION>Business</DESCRIPTION>
<LANGUAGEID>EN</LANGUAGEID>
<LOCATIONTYPECODE>BUS</LOCATIONTYPECODE>

</ADDRESS_LOCATIONTYPECODE>… Etc.

StateCode.xml

STATECODE>
<COUNTRYCODE>US</COUNTRYCODE>
<COUNTRY3CHARCODE>USA</COUNTRY3CHARCODE>
<DESCRIPTION>New York</DESCRIPTION>
<STATEPROVINCECODE>NY</STATEPROVINCECODE>
<DISCONTINUE>9999-12-31</DISCONTINUE>
<EFFECTIVE>1901-01-01</EFFECTIVE>
<LANGUAGEID>EN</LANGUAGEID>
<STATE3CHARCODE> </STATE3CHARCODE>
</STATECODE>
… etc.

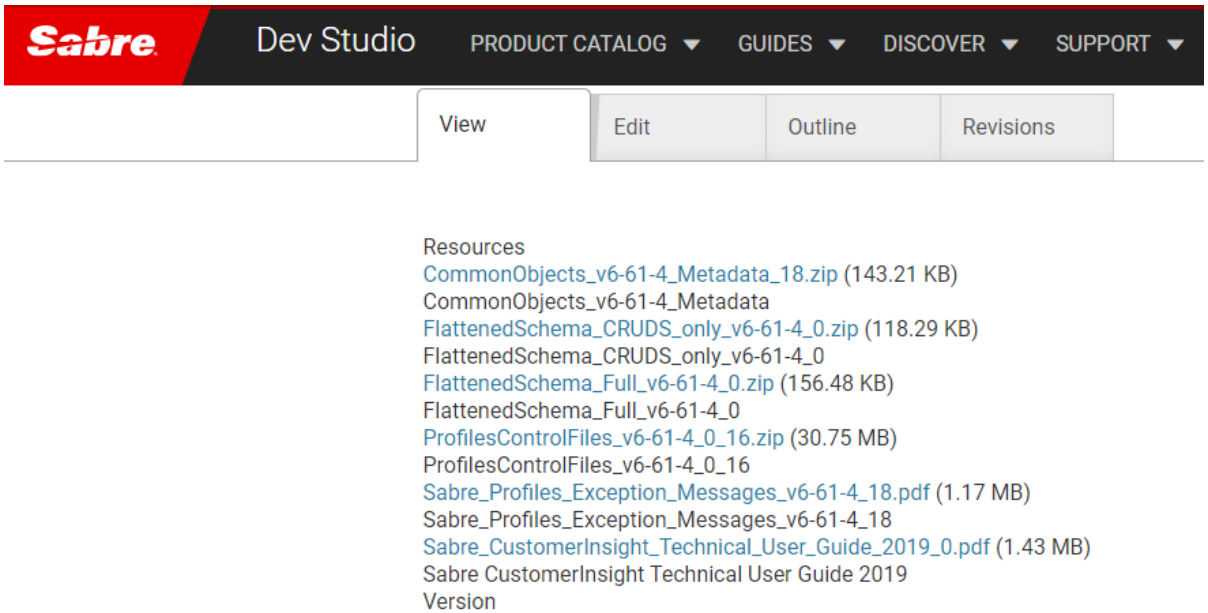
One particularity of the StateCodes is that logic in the system enforce the combined used of CountryCodes and StateCodes. This means that you can only use a StateCode that is valid for a given CountryCode. (ie. StateCode=”NY” is valid if CountryCode=”USA”, but not if CountryCode=”MEX”. It is also allowed to use StateCode only if CountryCode is provided in the xml. StateCode cannot appear in the xml if the CountryCode is missing.

Some of the most often used dictionary control tables include:

PROFILETYPECODE.xml, ADDRESS_LOCATIONTYPECODE.xml, AIRLINEPREF_TRIPTYPECODE.xml, AIRPORTCODE.xml, AIRPREF_MEALTYPECODE.xml, BANKCARDVENDORCODE.xml, COUNTRYCODE.xml, CUSTOMER_GENDER

CODE.xml, DOCTYPECODE.xml, EMAILTYPECODE.xml, PRIMARYLANGUAGEIDCODE.xml, SEATINFO_SEATPREFERENCECD.xml, SECURITYQUESTIONCODE.xml, SSRCODE.xml, STATECODE.xml, TELEPHONE_LOCATIONTYPECODE.xml

The complete list of XML / XSD files on control table (dictionary data) can be obtained from the Customer Insight resources on Sabre Developer Studio. Please find more information in previous section of this document.



From there, you must select the option to download ProfilesControlFiles_v6-61-4_0_16.

Note CI follows a naming convention ending the attribute name with "**Code**" to indicate that such element or attribute is governed by a dictionary control table. Therefore an easy way to identify from the schema or from a XML message which values are controlled by these Control Tables is to look for the suffix "Code" at the end of the string.

i.e.: … ProfileStatus**Code**="AC"
ProfileType**Code**="CRP"
…
<PriorityRemarks Category**Code**="ACC"
Text="Pr remark1"/>
<Remark Category**Code**="ACC" …. Etc.

These Dictionary / Control table files often follow the same naming convention as the respective elements in schema. For example, under the subject area *Telephone*, the element *LocationTypeCode* is controlled by the values set in **Telephone_LocationTypeCode.xml,** under *FormOfPayment-->PaymentCard* the attribute *CardTypeCode* is controlled by the values set in **CardTypeCode.xml,** the *ProfileStatusCode* attribute is controlled by the values set in **ProfileStatusCode.xml,** etc.

3.11 Use of Status codes

Access to *CI* for different clients is controlled by the “DomainID”.

For Sabre Airline Solutions clients, the domain is referenced by the 2 letter airline code for example: AA, AB, B6 .. LA,.., etc.

In order to be able to use DomainID, the CI development team must first have activated the “Domain” for the customer.

Profile “Status”

Profiles have Status element that determines the actions that are available, meaning status of profiles determins if profile can be updated or if can be read. The table below illustrates enabled and disabled functions according to the status of the profile.

Valid status for each operation below					
	AC	IN	DE	DL	MG
Create	OK	OK	OK	X	X
Update	OK	OK controlled by IgnoreStatusCheck attribute	OK require to ignore date/time stamp or read for updateDateTime	X	X
Read	OK	OK	X	X	X
Search	OK	OK	OK	OK (controlled by ExcludeDeleted attribute)	X
Delete/Restore	OK (set status DL)	OK (set status DL)	OK (set status DL)	OK (from status DL set status AC with **Restore** option)	OK

Note All status can be read / update if the payload is qualified with a special attribute to indicate to ignore the profile status. This condition should be used only for exception cases where client application needs to read / update the profile content for a profile with a status that in normal request would not be allowed. This is illustrated in the below example:

<Sabre_OTA_ProfileReadRQ

Target="Production" TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" xmlns="http://www.sabre.com/eps/schemas" xmlns:xsi="http://www.w3.org/2018/XMLSchema-instance" xsi:schemaLocation="http://www.sabre.com/eps/schemas ..\schemas\Sabre_OTA_ProfileReadRQ.xsd" **IgnoreStatusCheck=”Y”>**

3.12 Profile Associations

CI Profiles are not hierarchical, but rather all profiles can be viewed at the same level. However, Profiles can be associated to one another to indicate a relationship. The type of relationships can vary according to the customer's business needs. For example a traveler profile can be associated to a Corporate profile as the Traveler may work for the corporation or perform travel on their behalf.

Agent profile created via SSW is always associated to particular Agency profile on behalf of which they make bookings. That way the Agency IATA number is moved into PNR and incentives may be granted to the Agency.

SSW Corporate portal allows to associate Agency profile into Corporate. That would allow Agency which makes a booking on behalf of a Corporation to consume fare discounts granted to that Corporation.

A Traveler Profile can also be associated to one or more Travelers, etc.

The table below describes the possible ways inwhich the CI system allows profiles of a given type to be associated to other following the hierarchy as noted in the table.

FROM Profile Type	Allows Associations **TO** Profile Type
TVL	TVL, CRP, AGT, AGY
AGT	AGT, TVL, CRP, AGY
CRP	CRP, AGY, AGT
AGY	AGY, AGT

The table below describes standard profile associations used by Airline Solution customers using SabreSonic products.

FROM Profile Type	Allows Associations **TO** Profile Type
TVL	CRP
AGT	CRP, AGY
CRP	AGT, AGY

3.13 Common Rules and Elements across Profile Types

The table below illustrates high-level elements that are common across profile types and therefore also indicates under which section of the document details of those elements will be covered. That way, when the same element is used in a subsequent profile type, rather than duplicating all the information, the reader is asked to reference the corresponding profile type and section where these details have already been covered.

Element	TVL	AGT	CRP	AGY	ALC	Description
Customer	X					Parent element for Traveler under which the rest of the traveler elements can be found such as name, phone, address, etc.
PersonName	X				X	Title, First name, middle name , last Name, etc.
Telephone	X	X	X	X		All the components of a telephone, for full phone or parsed phone number (Country code, area code, number, etc.)
Email	X	X	X	X		Email address, usage, etc.
Address	X	X	X	X		Address lines, city, country, zip, etc.
PaymentForm	X	X	X	X		Form of payment type including Credit/Debit cards, Travel Bank account, checks, cash, accounts receivable etc.
RelatedIndividual	X					Contact data (names, etc.) for individuals related to the traveler profile.
EmergencyContactPerson	X	X	X	X		Contact information including names, phone numbers, email, address for contacting in case of emergency.
Document	X	X				Information such as passport, visas, driver licenses, etc.
CustLoyalty	X	X			X	Information on Loyalty programs per vendor such as airline frequent traveler programs, hotel or car rental frequent traveler or loyalty programs.
EmploymentInfo	X	X				Information about where the individual is employed including employee number, cost centers, departments, etc.
URL	X		X	X		URL address
AgencyContactName		X				Title, First name, middle name , last Name, etc.
AgencyInfo				X		Agency details including name, description, URL and other identifiers (also Tax and credit info)etc.
CorporateInfo			X			Corporation details including name, description, nature of business, and other identifiers (also Tax and bank info) etc.

Element	TVL	AGT	CRP	AGY	ALC	Description
GroupInfo						Group details including name, Meeting Id or Group Id (also Tax and bank info) etc. Note: Used only for Travel Network clients.
ContactName			X			Contains basic information about contact (name Prefix, GivenName, MiddleName, Surname (required), NameSuffix, PreferredFirstName, PreferredLastName, OrderSequenceNo, and LanguageIdCode).
AgentName		X				Agent name title, first name, middle name , last name, etc.
AgentInfo		X				Agent information such as date of birth, marital status, gender, etc. Also includes tax info.
AgentRelatedIndividuals		X				Data to identify other individuals related to the agent including their names and relationship type.
AgentGDSIdentity		X				Includes Agent sine, Id, GDS-code, etc.
PrefCollections	X	X	X	X		This section is divided by Air, Hotel, VehicleRental, Rail, etc.
AirlinePref	X	X	X	X		Stores preferences by trip type, geographic origin or destination and includes options for Airport preferences, seats, cabin, meals, preferred airlines, etc.
HotelPref	X	X	X	X		Stores preferences by trip type, geographic origin or destination and includes options for preferred hotel vendors, property, rates, roomType, etc.
VehicleRentalPref	X	X	X	X		Stores preferences by trip type, geographic origin or destination and includes options for preferred rental car vendors, rates, vehicle types, etc.
RailPref	X	X	X	X		Stores preferences by trip type, geographic origin or destination and includes options for rail stations, seats, cabin, meals, vendors, upgrades, etc.

Element	TVL	AGT	CRP	AGY	ALC	Description
GroundTransportationPref	X	X	X	X		Stores preferences by trip type, geographic origin or destination and includes options for preferred vendors, rates, aggregators, etc.
TPAMarkertingPreference	X	X	X	X		Includes notification preferences, consent for notifications and agreement for campaigns and consent for terms and conditions, etc.
LoungePref	X					Stores preferences by trip type, and vendor. Includes options preferred drink, preferred newspaper etc.
Priority Remarks	X	X	X	X		Important notes / information.
Remarks	X	X	X	X		Other notes / remarks.
SSR	X					Special Service Request information including codes and descriptions (i.e. WHCR - Wheel chair), etc.
OSI	X					Other Service Request information including codes and descriptions.
CustomerReferenceInfo	X	X	X	X		Reference to 3rd party systems such as accounting systems including branch and account ID's for DK-customer storage.
BusinessSystemIdentityInfo	X	X	X	X		Information on other systems (i.e. back office system references) includes system name, Id, description, etc.
Associated Profiles	X	X	X	X		To reference other profiles associated to the current profile (includes, profile ID, profile associated type codes, DomainID (PCC), and information on whether the associated profile data should move into a PNR before or after the main profile.
Discounts	X	X	X	X		Discount information by vendor for Air, Hotel, Car Rental discounts, etc. (includes vendor type and code, discount number, descriptions, effective and expiration dates) etc.

Element	TVL	AGT	CRP	AGY	ALC	Description
CustomDefinedData	X	X	X	X		Value pair data including codes, descriptions and data values to allow customers to define their own data beyond the existing data model. Validation applied by Profiles System for codes sent against those pre-defined for domain via DataSrvRQ.
CustomDefinedValues	X	X	X	X		String values defined by customer to define their own data beyond existing data model. No validation applied by Profiles system for content.
Tax Info	X		X	X		Tax information corresponding the entity in the Profile.
Roles		X				Reference to the Role assigned (grouping of lower level permissions) to enable / restrict profile functions such as Create, Read, Update, Delete, etc. Note: Used only for Travel Network clients.
Travel Policy	X		X	X		Reference to Sabre Corporate Travel Policy records (CTP / Flex edits) and Sabre Travel Policy Engine Note:Used only for Travel Network clients.
STARData	X	X	X	X		Reference to the Sabre STAR data from which the profile originated (for profiles migrated from STARs). Note:Used only for Travel Network clients
ProfileAccessInfo	X	X	X	X		Read only fields to track profile usage (create and update).
CustomerValueScore	X					Customer value score assigned by a system to segment customers (includes, score, effective / discontinue dates, vendor types and codes).
GDS		X	X	X		Information of the Global distribution system referenced to this profile (I.e. 1S=Sabre).
QueueAssignments		X		X		Queue assignment including queue number, DomainID (PCC) Prefatory instruction, etc.

Element	TVL	AGT	CRP	AGY	ALC	Description
Commissions		X		X		Commission data including: commission measure code, value, trip type, vendor, effective/discontinue date, etc.
Incentives		X				Information on incentives including: supplier code, service, incentive ID, value, effective/expiration dates, etc.
EmploymentInfo	X	X	X	X		Information on employment including employee id, title, cost center, division, department, company, business unit, etc.
SocilalMediaAccounts	X	X	X	X		Stores social media accounts information like username
AnalyticalInfoGroup	X					Stores analytical information assigned by a systems to segment customer like customer value score

Notice that there could be a number of Subject Areas and Elements that even though are present in the generic data model might not be utilized by Airline Solutions customers and are intended to be used by other business areas such as Travel Network.

3.14 Common Elements and Attributes

All **ProfileCreateRQ** requests require the following elements and attributes: *UniqueID*, *ProfileType*, *ClientCode*, *ClientContextCode* and *DomainID.*

In case of UniqueID a value can be provided by a client / user, or if set to "*" a unique identifier will be generated by Customer Insight. In case where the airline client has specified the requirement for the Loyalty number to be set in a specific number range and / or following a specific validation algorithm (i.e. MOD 7, MOD 10 etc.). the UniqueID will be generated according to those set rules. The UniqueID will be generated according to rules set up in CI but the number will be generated randomly.

ProfileTypeCode (TVL for Traveler, CRP for Corporate, AGT for Agent, AGY for Agency, ALC for Alliance) is a mandatory element. The ClientContextCode is also mandatory and is used to define the application that is creating the profile. (i.e. SSW for SabreSonicWeb, INT for Interact, FLC for Flight Compensation Tool etc.). However, when authorized developer customers are using *CI* services to create profiles, they should use Client Context Code "EXT" (External Application). The values for this attribute are managed by the dictionary of control values XML.

The values of the ClientCode and the ClientContextCode are related to each other and they cannot be set separately. The matching ClientCode for the ClientContextCode is also specified on Sabre Dev Studio in the STNDRDPRFRLS_CLIENTCNTXTCD.xml.

The DomainID should be set as the 2 characters airline code for Airline Solution customers.

Both LoginID and UniqueID must be set as unique identifiers for a profile across all profiles within a given DomainID. For example, if Traveler (TVL) profile with LoginID = User1234 already exists, then a user who creates an Agent (AGT) profile must set a LoginID different from LoginID="User1234". Otherwise the following error massage is displayed: " *Profile with Login ID User1234 already exists for Client Code 'AS' and Domain ID 'MN'* ".

The element "OrderSequenceNo" should be used when the user needs to specify the particular order within an element that has 2 or more instances (i.e. 2 or more phone numbers, etc.).

Various sections of the schema require an attribute *LanguageIDCode* which if not specified,will be set to the default value which is 'EN-US'.

During profile creation request if the ProfileStatusCode is not specified then *CI* application will create a profile in Active (AC) status by default.

Please review carefully the "Traveler Profile" section below, as the majority of the rules and elements available will also apply to other profile types, but details will be covered only under this section to avoid duplication of information.

Note Any time the content of the elements for a profile needs to include special characters that are used in XML formatting, you need to replace these characters with their corresponding value as shown in the table below:

<	&lt;
&	&amp;
>	&gt;
"	&quot;
'	&apos;

Management of Non-standard English Characters

CI allows storage and management of double-byte characters and supports special characters including Indian/Chinese Double Byte & Arabic ASCII characters in XML, and also for special Sabre characters Such as the "Cross of Loraine", "Change Key", and "End Item".

All Sabre special characters are stored as Unicode values:

- Cross of Loraine ¥ : ¥
- Change key ¤ : ¤
- End item § : §

Entries	Display in a web page
<?xml version="1.0" encoding="UTF-8"?> <Test> <Test1>ఢ</Test1> <Test1>§</Test1> <Test1>乖</Test1> <Test1>ڄ</Test1> </Test>	<?xml version="1.0" encoding="UTF-8" ?> <Test> <Test1>ఢ</Test1> <Test1>§</Test1> <Test1>乖</Test1> <Test1>ڄ</Test1> </Test>

3.15 Profile UniqueID

In *CI*, all profile types include an attribute under TPA_Identity, called “UniqueID”. This ID uniquely identifies the profile and the number cannot be changed.

The Profile ***UniqueID*** might be set up by the client or it can be automatically generated (according to the pre-defined range number and algorithm selected by the airline) by the *CI* system to ensure uniqueness across all profiles within a Domain. This UniqueID is entered into the PNR when a Profile is moved to create a PNR in a section called “Profile Index” (PI).

The PI field of the PNR is what links profiles to PNRs and ties this data together. However, this profile index does not include a reference to the DomainID which owns the profile from where it was moved. The UniqueID used for the Index is unique across all profiles within a Domain.

If the Profile UniqueID is auto-generated, no specific value should be provided for this field when creating a profile. The value of “*” should be provided for this attribute in the CreateRQ. To generate the UniqueID automatically, the profile create request should look like this:

```
<Profile CreateDateTime="2001-12-17T09:31:47.0Z"
            UpdateDateTime="2001-12-17T09:30:47.0Z"
  PrimaryLanguageIDCode="EN-US">
<TPA_Identity UniqueID="*" ProfileTypeCode="TVL" ClientCode="AS"

    ClientContextCode="EXT"   ProfilePurgeNoDays=”5”
DomainID="XX"
         ProfileName="John123"  ProfileStatusCode="AC"
         ProfileDescription="Travelr sample">
</TPA_Identity>
```

Create Service 4

4.1 Creating Profiles

CI web services clients create a customer profile over the web interface using the **Sabre_OTA_ProfileCreateRQ** request.

Also refer to the Sabre Dev Studio for a detailed payload illustrating the data elements and attributes available in xml files. Each file is annotated to explain data restrictions, whether optional or required data, and other detailed explanations.

The following sections describe the most typical subset of data currently in use.

4.2 How to Create a Profile

The below information uses Traveler profile as example in the xml.
*Assume the **SessionCreateRQ** has already returned the binary session token and conversation ID for the web client session being described.*

4.3 Sample XML Create Profile Request

The Create XML request begins with the following boilerplate:

<Sabre_OTA_ProfileCreateRQ TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production" xmlns="http://www.sabre.com/eps/schemas" >

In the opening Sabre_OTA_Profile_CreateRQ payload the following optional attributes might be set which will trigger certain actions in the CI system. Below a list of attributes frequently used by Airline Solution customers:

- **ProfileDuplicateCheck** – if set up to “Y” in a payload then *CI* will check if duplicated profile (or possible duplicates) may already exist in *CI*. Duplicate check process follows the duplicate check pattern selected by customer and set up in *CI* database at the time of activation of the client domain. Available patterns include: name and email, name and address, name and phone or name and date of birth. If ProfileDuplicateCheck attribute is not populated, then by default it is set up as “N”. If duplicate check is not configured then this attribute has no effect.
- **GenerateLoginID** – is used to generate login by *CI*. If payload contains a UniqueID=”*” and GenerateLoginID=”Y” then *CI* will generate both UniqueID and LoginID with the same value which follows the algorithm provided by the customer and is set up in *CI* database. If payload is sent with dedicated UniqueID and flag GenerateLoginID is set to “Y” then *CI* will generate LoginID equal to UniqueID sent by customer in the payload. LoginID must be unique for each domain.
- **GeneratePassword** – is used to generate password by *CI*. If set to “Y” then *CI* will generate random unique numeric password which follows the rules (password length) provided by customer and set up in *CI* database. If *CI* should generate a password, then the CreateRQ payload should also include LoginID information, whether set up to auto generate the “LoginID” (GenerateLoginID=“Y”) or LoginID value must be set up by customer and included in the payload.

- **GenarateMembershipID** – is used to generate <CustLoyalty> section where the Frequent Flyer membership of the airline loyalty system is stored. If GenerateMembershipID is set up as “Y” and UniqueID=“*” then *CI* will generate value according to algorithm provided by customer and set up in *CI* database with same content for both UniqueID and CustLoyaltyàMembershipID. If UniqueID is specified in a payload and GenarateMembershipID is set to “Y” then *CI* will add CustLoyalty section with MembershipID with same value as sent in a payload for UniqueID. If value GenerateMembershipID is not included in the payload then *CI* will not create CustLoyalty element. Whenever UniqueID=”*” and the ID generation algorithm was set, UniqueID value will be autogenerated. There is no need for a dedicated attribute flag that controls this.

Note Customers have a possibility to create profiles with passwords in form of their hash accordingly to the agreed upon hash algorithm, otherwise passwords are kept encrypted in the *CI* database. Keeping hash of a password prevents the original passwords from being retrieved if forgotten or lost so that they must be replaced with new ones.

```
<Sabre_OTA_ProfileCreateRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas"
ProfileDuplicateCheck="Y"
GenerateLoginID="Y"
GeneratePassword="Y"
GenerateMembershipID="Y">
```

The next section is the element **Profile** with the following attributes: *CreateDateTime, UpdateDateTime* and *PrimaryLanguageIDCode*. The first two of these attributes are required and the last one is optional. If the value of the *PrimaryLanguageIDCode* is not set, it will be set to “EN-US” by default. The *PrimaryLanguageIDCode* can hold one of the values specified in the Dictionary Control Value XML files. The xml file name for the control table values available is PRIMARYLANGUAGEIDCODE.xml.

```
<Profile CreateDateTime="2018-11-17T09:31:47.0Z"
          UpdateDateTime="2018-11-17T09:30:47.0Z"
  PrimaryLanguageIDCode="EN-US">
```

Now define the unique identification for this profile in the TPA_Identity section:

```
<TPA_Identity ClientCode="AS" ClientContextCode="EXT"
   UniqueID="*" ProfileTypeCode="TVL"
DomainID="XX"  ProfileStatusCode="AC">
 </TPA_Identity>
```

The following attributes: *UniqueID, ProfileType, ClientCode, ClientContextCode* and *DomainID* are required. In the example above, the *UniqueID* has been set to “*” which means that a valid unique identifier will be generated on the database side. The *ProfileTypeCode* has been set to “TVL” which means that the profile to be created is of the Traveler type. Apart from ‘TVL”, the *ProfileTypeCode* can be set to one of the following values: “AGT”, “AGY”, “CRP”. They can be defined respectively as: Agent, Agency, Corporate type of profiles. These values are stored in **ProfileTypeCode.xml** (as part of the Dictionary Control Table values referenced in Sabre Dev Studio).

The *ClientCode* should be set to “**AS**” for Sabre Airlines Solutions customers.

The *ClientContextCode* is mandatory and should be set to “EXT” for *Sabre Airline Solutions* customers connecting to *CI*. Other values for other clients are specified in **TPA_Identity_ClientContextCode.xml** (as part of the Dictionary Control Table values referenced in Sabre Dev Studio).

However, the values of the *ClientCode* and the *ClientContextCode* are related to each other and they cannot be set separately. The matching *ClientCode* for the *ClientContextCode* is stored in: **TPA_Identity_ClientContextCode.xml.**

<CLIENTCONTEXTCODE>**EXT**</CLIENTCONTEXTCODE>
<DESCRIPTION>**External AS Application**</DESCRIPTION>
<APPLICATION>**AS**</APPLICATION>

The *DomainID* should be a 2-character airline code for which the Profile operation is performed.

The rest of attributes (ProfileName, ProfileNameModifyIndicator, ProfileDescription, ProfileStatusCode and ProfilePurgeNoOfDays) are optional.

If the value of any of the mandatory attributes is set as an empty string or not set at all, the user will receive an appropriate error message.

Note It is important to remember not to add an element or attribute that does not contain any value or empty value or white spaces. For example, white spaces at the end of a surname might cause that profile cannot be found when searching by name.

4.4 Sample XML Create Profile Successful Response

When a profile is successfully created, the following response message is returned:

```
<Sabre_OTA_ProfileCreateRS xmlns=”http://www.sabre.com/eps/schemas" Version="6.49"
TimeStamp="2018-12-06T14:07:11.388Z" CreateDateTime="2018-12-06T14:07:11.388Z">
        <ResponseMessage>
                <Success />
        </ResponseMessage>
        <Profile ProfileID="123456789" ProfileType="TVL" ClientCode="AS"
ClientContextCode="EXT" DomainID="XX">
        </Profile>
</Sabre_OTA_ProfileCreateRS>
```

The Response message contains a Profile section, which returns a subset of information about the newly created profile. This information is sufficient to retrieve this profile from *CI* database.

4.5 Sample XML Create Profile Error Response

When a profile cannot be created for some reason, an error message is returned. Each error message contains an Errors section with an Error Code and a short description of the problem which was encountered during the profile creation. For the Sabre Profile Create Service each Error message is prefixed with “C::”.

In the example below the profile could not have been created because in the *CI* database a profile already exists with the same credentials (with the same LoginID attribute value). The error message has been prefixed with “C::”

<Sabre_OTA_ProfileCreateRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-10-01T14:47:49.607Z" Version="6.49">
  <ResponseMessage>
    <Errors>
      <ErrorMessage ErrorCode="80"> **C::NCERT502-1T20131001144749SC::Profile with Login ID**
      3638709760 already exists for Client Code 'AS' and Domain ID 'XX'</ErrorMessage>
    </Errors>
  </ResponseMessage>
  <Profile ClientCode="AS" ClientContextCode="EXT" UniqueID="*" ProfileTypeCode="TVL"
DomainID="XX" ProfileStatusCode="AC">
    <Login LoginID="3638709760" />
  </Profile>
</Sabre_OTA_ProfileCreateRS

For examples of Error Messages, see Sabre Dev Studio under *CI Resources*.

Read Service 5

5.1 Reading Profiles by UniqueID

Profiles customer can be retrieved by specifying the ProfileID, ClientContextCode, ClientCode, DomainID and ProfileType.

A sample Profile Read Request looks like this:

```
<Sabre_OTA_ProfileReadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" >
                <Profile>
        <TPA_Identity ProfileID="4250" ProfileType="TVL" ClientCode="AS" DomainID="XX"
ClientContextCode="EXT">
        </TPA_Identity>
                </Profile>
                </Sabre_OTA_ProfileReadRQ>
```

5.2 Reading Profiles by LoginID

Profiles customer can be retrieved by specifying the LoginID, ClientContextCode, ClientCode, DomainID and ProfileType.

A sample Profile Read by Login looks like this:

```
<Sabre_OTA_ProfileReadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas">
        <Profile>
                <TPA_Identity DomainID="XX" ProfileTypeCode="TVL" UniqueID="*"
ClientContextCode="EXT" ClientCode="AS">
<Login LoginID="smithowski@example.com"/>
                </TPA_Identity>
        </Profile>
</Sabre_OTA_ProfileReadRQ>
```

5.3 Authenticate customer by Login and password

Customer Insight has ability to authenticate customer when LoginID and password are provided in the Read request. It is not necessary to indicate the ProfileTypeCode in such a request.

If the credentials specified in the request match what is stored in the *CI* profile, then this profile is displayed in Read response. If the password provided in the xml does not match what is in *CI*, then error massage is displayed.

Xml request below illustrate the scenario:

```
<Sabre_OTA_ProfileReadRQ
```

```
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" xmlns:xsi="http://www.w3.org/2018/XMLSchema-
instance"
                  <Profile>
                 <TPA_Identity DomainID="XX" ProfileTypeCode="TVL" UniqueID="*"
ClientContextCode="EXT" ClientCode="AS">
                          <Login IsHashed="N" PasswordHash = "abcd1234"
LoginID="smithowski@example.com"/>
                 </TPA_Identity>
         </Profile>
</Sabre_OTA_ProfileReadRQ>
```

Xml error massage if credentials do not match:

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-10-25T11:00:01.446Z" Version="6.49">
 <ResponseMessage>
  <Errors>
   <ErrorMessage ErrorCode="53">
   R::NCERT501-2T20131025110001SR::Password Credentials are invalid</ErrorMessage>
  </Errors>
 </ResponseMessage>
</Sabre_OTA_ProfileReadRS>
```

Read response contains the latest version of the profile stored in CI.

Using **Sabre_OTA_ProfileReadRQ** user can check the credentials of the user that made recent changes to a profile. To receive such information, the optional attribute ReturnAuditInfo set to “Y” must be included in the payload.

```
<Sabre_OTA_ProfileReadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" xmlns:xsi="http://www.w3.org/2018/XMLSchema-
instance" ReturnAuditInfo="Y">
 <Profile>
  <TPA_Identity ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
ProfileTypeCode="TVL" UniqueID="6560015693"/>
 </Profile>
</Sabre_OTA_ProfileReadRQ>
```

The example below illustrates the Read response that contain <AuditInfo> section showing information about last modification

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas" Version="6.49">
 <ResponseMessage>
  <Success />
 </ResponseMessage>
 <Profile CreateDateTime="2018-10-29T07:54:53.071Z"
 UpdateDateTime="2013-09-18T09:44:26.978Z"
 PrimaryLanguageIDCode="EN">
  <TPA_Identity ClientCode="AS" ClientContextCode="EXT"
  UniqueID="6560015693" ProfileTypeCode="TVL" DomainID="XX"
```

```
   ProfileStatusCode="AC">
     <ProfileSubType SubTypeCode="FFP" />
   </TPA_Identity>
 <Traveler>
                ...        < All Traveler information>
...
  <AuditInfo>
   <CreateAccessInfo EPRDomainID="UPLOAD" DutyCd="Mock"
   UserID="Mock" LDAPDomain="UPLOAD" LNIATA="000000"
   AgentSine="Mock" AgentGivenName="Mock" AgentMiddleName="Mock"
   AgentSurName="Mock" Timestamp="2018-10-29T07:54:53.071Z" />

   <LastUpdateAccessInfo EPRDomainID="Employees" UserID="205438"
   LDAPDomain="Sabre" LNIATA="0"
   Timestamp="2018-09-18T09:44:26.978Z" />
  </AuditInfo>
</Sabre_OTA_ProfileReadRS>
```

Note In the example above, the section ”CreateAccessInfo” shows that the profile was created via “batch file upload” and therefore there is no specific user ID tracked for the transaction. In this case the system sets the EPRDomainID value as “UPLOAD” and user elements with word= “Mock”. However, if the transactions are processed by a client using web services, the user credentials performing the transaction are registered. This is shown in the “LastUpdateAccessInfo” from the above sample where a Sabre employee updated the record.

Note In the example above, the element “AuditInfo” together with ”CreateAccessInfo” and/or “LastUpdateAccessInfo” might not appear if the audit data was too old and therefore audit entries were removed and are no longer available

Another example illustrating a profile created by an airline agent is shown below:

```
<AuditInfo>
    <CreateAccessInfo EPRDomainID="XX" DutyCd="5" UserID="123123"
    LDAPDomain="XX" LNIATA="3155155" AgentSine="SSW"
    Timestamp="2018-06-21T04:08:36.875Z" />
   <LastUpdateAccessInfo EPRDomainID="AZO" DutyCd="5"
   UserID="123123" LNIATA="1DF1D1" AgentSine="ABC"
   AgentGivenName="AB" AgentSurName="SURNAME NAME"
   Timestamp="2018-11-08T12:18:38.855Z" />
 </AuditInfo>
```

Sabre_OTA_ProfileReadRQ also has the capability of reading specified subject areas and ignoring specified subject areas.

The <**PartialReadSubjectAreas**> element under **<Profile>** indicates which subject areas are of interest in the profile. If a profile has subject area(s) specified in **<PartialReadSubjectAreas>**, then the Profile read response returns only that subject area(s). It ignores all other subject areas in the profile. If the profile does not have a corresponding subject area specified in **<PartialReadSubjectAreas>**, then the Profile read returns only <TPA_Identity> section in response.

Here is the Sample Profile read request with **<PartialReadSubjectAreas>**:

```
<Sabre_OTA_ProfileReadRQ xmlns="http://www.sabre.com/eps/schemas" Version="6.49">
    <Profile>
        <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="100001069"
ProfileTypeCode="TVL" DomainID="XX"/>

    <PartialReadSubjectAreas><SubjectAreaName>PersonName</SubjectAreaName></PartialRe
adSubjectAreas>
    </Profile>
</Sabre_OTA_ProfileReadRQ>
```

<IgnoreReadSubjectAreas> indicates which subject areas are not of interest in the profile read response. In this case, the Profile read returns all subject areas except subject areas mentioned in **<IgnoreReadSubjectAreas>**.

Here is a Sample Profile read request with **<IgnoreReadSubjectAreas>**.

```
<Sabre_OTA_ProfileReadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas"
    <Profile>
        <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="100001069"
ProfileTypeCode="TVL"DomainID="XX"/>
        <IgnoreReadSubjectAreas>
<SubjectAreaName>PaymentForm</SubjectAreaName>
<SubjectAreaName>Remark</SubjectAreaName>
</IgnoreReadSubjectAreas>
    </Profile>
</Sabre_OTA_ProfileReadRQ>
```

5.4 Sample XML Read Successful Response

When a profile is successfully retrieved, the following response is returned.

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"
Version="6.49">
  <ResponseMessage>
    <Success />
  </ResponseMessage>
  <Profile CreateDateTime="2013-10-25T08:39:14.613Z"
  UpdateDateTime="2018-10-25T08:39:14.613Z"
  PrimaryLanguageIDCode="EN-US">
    <TPA_Identity ClientCode="AS" ClientContextCode="CI"
    UniqueID="3126384971" ProfileTypeCode="TVL" ProfileNameModifyIndicator="Y"
DomainID="XX" ProfileStatusCode="AC">
      <Login LoginID="email.as.a.login.or.ffnum.as.login@example.com"
PasswordHash="paswordinplaintextifnotinhash"  PasswordExpired="N" IsHashed="N" />
    </TPA_Identity>
    <Traveler>
                                    <!--Traveler information-->
```

```
   </Traveler>
  </Profile>
</Sabre_OTA_ProfileReadRS>
```

The most important section is the Profile section which contains the retrieved Profile. If the Profile was created but there were no updates made to this profile, then the CreateDateTime and the UpdateDateTime attributes contain the same timestamp.

If the retrieved profile contains credit card data in the Payment Form subject area, it can be shown in different ways depending on the EPR Keywords of the agent which were used to create the session.

If the agent account (EPR) does not have the Keyword ‘CCVIEW’ assigned (or if the OSCCVIEW security attribute is not assigned) then the credit card number and Travel Bank number stored in Payment Form within the retrieved profile will contain CardNumber="NOACCESS" and CCViewAccess="N".

A sample response looks like this:

```
 <PaymentForm DisplaySequenceNo="1" OrderSequenceNo="1" TripTypeCode="LS"
ServiceUsageTypeCode="AL" InformationText="PaymentForm">
                <PaymentCard CardTypeCode="CR" BankCardVendorCode="BA"
CardNumber="NOACCESS" MaskedCardNumber="1111" CCViewAccess="N"
EffectiveDate="012010" ExpireDate="012015" FirstRemark="N"
ExtendedPaymentNumMonths="5">
                  <CardHolderName>
                    <CardHolderFullName>JOHN DOE</CardHolderFullName>
                    <NamePrefix>MR</NamePrefix>
                    <GivenName>JOHN</GivenName>
                    <MiddleName>DALLAS</MiddleName>
                    <SurName>DOE</SurName>
                    </CardHolderName>
                  <CardIssuerName IssueNumberText="1234" IssuerName="DALLAS"/>
                </PaymentCard>
 </PaymentForm>
```

If the agent’s account (EPR) has the ‘CCVIEW’ Keyword or the OSCCVIEW security attribute assigned, the CardNumber or Travel Bank number stored in *CI* profile will be returned in the Read response:

```
              <PaymentForm DisplaySequenceNo="1" OrderSequenceNo="1" TripTypeCode="LS"
ServiceUsageTypeCode="AL" InformationText="PaymentForm">
                <PaymentCard CardTypeCode="CR" BankCardVendorCode="BA"
CardNumber="371111111111111" MaskedCardNumber="1111" CCViewAccess="Y"
EffectiveDate="012010" ExpireDate="012015" FirstRemark="N"
ExtendedPaymentNumMonths="5">
                  <CardHolderName>
                    <CardHolderFullName>JOHN DOE</CardHolderFullName>
                    <NamePrefix>MR</NamePrefix>
                    <GivenName>JOHN</GivenName>
                    <MiddleName>DALLAS</MiddleName>
                    <SurName>DOE</SurName>
                   </CardHolderName>
```

```
        <CardIssuerName IssueNumberText="1234" IssuerName="DALLAS"/>
      </PaymentCard>
    </PaymentForm>
```

If the user account has BTVIEW keyword or BTFOPView security attribute assigned to his web services user account (web services EPR), then Travel Bank account number will be returned in a Read response but Credit Card data will still be masked with “NOACCESS” as follows: CardNumber="NOACCESS" and CCViewAccess="N".

Note A Travel Bank account is stored under the Payment form subject area in similar manner to a Credit Card, with the difference that CardTypeCode is set to "TB" and BankCardVendorCode is set to "BT". These codes identified the form of payment as a Travel Bank account. System that reads PaymentForm elements without any need and without necessary keyword or security attribute, might easily overwrite a valid account number with “NOACCESS” value when making updates to the profile once read.

Example below illustrate Read response when user has BTFOPView attribute (and does not have neither OSCCVIEW nor CCVIEW) assigned to his account:

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"
Version="6.49">
 <ResponseMessage>
  <Success />
 </ResponseMessage>
 <Profile CreateDateTime="2012-11-29T07:54:53.071Z" UpdateDateTime="2018-09-
18T09:44:26.978Z"
 PrimaryLanguageIDCode="EN">
  <TPA_Identity ClientCode="AS" ClientContextCode="SSW" UniqueID="6560015693"
ProfileTypeCode="TVL" ProfileNameModifyIndicator="Y" DomainID="XX"
ProfileStatusCode="AC">
   <ProfileSubType SubTypeCode="FFP" />
  </TPA_Identity>
<Traveler>
 <Customer>
    <PaymentForm DisplaySequenceNo="1" OrderSequenceNo="1">
     <PaymentCard CardTypeCode="TB" BankCardVendorCode="BT"
     CardNumber="8795043193492642" MaskedCardNumber="2642" CCViewAccess="Y">
      <CardHolderName>
       <CardHolderFullName>Travel Bank Name</CardHolderFullName>
      </CardHolderName>

      <CardIssuerName IssuerName="Sabre Travel Bank" />
     </PaymentCard>
    </PaymentForm>

    <PaymentForm DisplaySequenceNo="2" OrderSequenceNo="2">
     <PaymentCard CardTypeCode="CR" BankCardVendorCode="BX"
     CardNumber="NOACCESS" MaskedCardNumber="1111" CCViewAccess="N"
EffectiveDate="012010" ExpireDate="012015">
      <CardHolderName>
```

```
          <CardHolderFullName>Test User</CardHolderFullName>
         </CardHolderName>
        </PaymentCard>
       </PaymentForm>
     </Customer>
   </Traveler>
  </Profile>
</Sabre_OTA_ProfileReadRS>
```

If a user has neither BTFOPView nor OSCCVIEW security attribute assigned to his TPF user account or open systems EPR, both Credit Card and Travel Bank number will be returned in read response containing CardNumber="NOACCESS" and CCViewAccess="N".

If a profile contains associated profiles, they are returned in the Profile Read response as well. A sample ProfileRead response with associated profiles is shown below:

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"
Version="6.49">
  <ResponseMessage>
    <Success />
  </ResponseMessage>
  <Profile CreateDateTime="2013-10-07T08:27:05.608Z"
  UpdateDateTime="2018-10-07T08:27:05.608Z"
  PrimaryLanguageIDCode="EN">
    <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="30831022"
ProfileTypeCode="TVL" ProfileNameModifyIndicator="Y" DomainID="XX"
ProfileStatusCode="AC">
      <Login LoginID="TESTUSER123" PasswordHash="abcd1234" PasswordExpired="N"
IsHashed="N">
      </Login>
    </TPA_Identity>
    <Traveler>
      <TPA_Extensions>
        <AssociatedProfiles AssocUniqueID="13904713" AssocProfileTypeCode="CRP"
DisplaySequenceNo="1" OrderSequenceNo="1" DomainID="XX" ClientCode="AS"
ClientContextCode="EXT"  ProfileRelationTypeCode="EM" ProfileRelationStatusCode="AC"
CreditBankIndicator="N"/>

        <AssociatedProfiles AssocUniqueID="13904713" AssocProfileTypeCode="CRP"
DisplaySequenceNo="2" OrderSequenceNo="2" DomainID="XX" ClientCode="AS"
ClientContextCode="EXT"  ProfileRelationTypeCode="TP" ProfileRelationStatusCode="DE"
CreditBankIndicator="N"/>

        <NumberOfAssocProfiles Corporation="2" />
      </TPA_Extensions>
    </Traveler>
  </Profile>
</Sabre_OTA_ProfileReadRS>
```

The **<NumberOfAssocProfiles>** element indicates the total number of associations attached to the profile. This includes all associations that the profile being read is associated to and the number of

profiles associated profiles to that profile. The total count of Traveler, Travel Agent, Travel Agency, Corporation profiles is included in this section.

5.5 Sample XML Read Error Response

When a profile cannot be read for some reason, an error message is returned. Each error message contains an Errors section with an Error Code and a short description of the problem which was encountered during the profile read process. For the *CI* Read Service, each Error message is prefixed with “R:::”. A sample error message is shown below:

```
<Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2013-10-07T07:52:58.207Z" Version="6.49">
  <ResponseMessage>
    <Errors>
      <ErrorMessage ErrorCode="298">R::NINT1T20131007075258SR::No  profile is found which
match your selection criteria   (UniqueID = 1234, ClientCode = AS, ClientContextCode = EXT,
     DomainID = XX, ProfileTypeCode = AGY, LoginID = null)</ErrorMessage>
    </Errors>
  </ResponseMessage>
</Sabre_OTA_ProfileReadRS>
```

In the above example, the Profile which was supposed to be retrieved, does not exist in the *CI* database.

Update Service 6

6.1 Updating Profiles

A customer can update a profile in one of two ways. A profile can be completely updated by full replacement of data, or partially updated. Partial update functionality allows to delete a specific element, add an element or change it.

To fully update a traveler profile, exact value of *UpdateDateTime* is needed. It can be retrieved with a **ProfileRead** service or captured from the previous update from **ProfileUpdateRS.**

Timestamps are generated by *CI* system using the date and time in GMT time zone.

1. In order to properly update a profile, a system that needs to make this update must perform some preliminary steps in order to correctly perform an update. The system is named “web client” further in this description. Once a web client has a complete profile information ready to be updated, it either uses UpdateDateTime value captured after making previous update or needs to read a profile first with ProfileRead. Web client can also merge anychanged data with the unchanged data that was previously retrieved from CI system, in that case web client sends the merged data to *CI* and uses the timestamp retrieved from ProfileRead response. The *CI* application validates the UpdateDateTime timestamp value to maintain data integrity. If the same or another web client updates the profile prior to the receipt of the update from this client, the timestamp value of UpdateDateTime would not match the timestamp already updated in the *CI* database and an error response would be returned indicating an error code “302” with the following message: "Invalid Last Update Time Stamp. Profile probably has been updated by another client”
2. In such situation web client needs to retrieve the profile again, merge the changes again, and resend the update.

Note *CI* supports an exception to validation described above where the client application sending the update request can set an optional parameter “IgnoreTimeStampCheck= “Y”, which allows to conduct the update ignoring the timestamp from the request. Although this prevents the user from the need of performing a Profile Read and obtaining the latest update flag, this option should be used with care and is only recommended when a single source is used to update the profile and is certain that the information sent in the update is the master data that should be retained.

To partially update a customer profile no prior retrieval is necessary, although in certain cases old data needs to be known and included in the update request. The web client sends only the customer profile data type that needs to be changed to *CI*. For example, if only an email is changed, then only the email related data is sent. Internally *CI* maintains data integrity by locking the profile data during processing.

In the case of a partial update, three possibilities exist.

The first is related to an update of a specific element, which relies on matching existing data and changing it to new data. The second is related to the addition of a new element to a specified path (branch), and the last one with the deletion of an element, which was indicated to be deleted.

All the above operations can be put in one **PartialUpdateRQ**, and additionally, each one of them (delete, add, modify) can contain up to 200 single changes.

Note Although the most often used subject areas and data elements have been implemented to support partial updates, the entire data model is not supported. Please validate with CI implementation team the data elements you would like to implement via “partial updates”. If your data set includes areas that are not yet supported by this method, you would have to use “Full Update”.

The following Subject Areas currently support Partial Updates:

- CUSTOMER-ATTRIBUTES
- PERSONNAME
- ADDRESS
- EMAIL
- TELEPHONE
- PAYMENTFORM
- RELATEDINDIVIDUAL
- EMERGENCYCONTACTPERSON
- DOCUMENT
- CUSTLOYALTY
- EMPLOYMENTINFO
- CUSTOMDEFINEDDATA
- SSR
- OSI
- CUSTOMERREFERENCEINFO
- DISCOUNTS
- CUSTOMERVALUESCORE

6.2 How to update a profile using full replace

The full traveler profile update request is shown below:

```
<Sabre_OTA_ProfileUpdateRQ Target="Production" TimeStamp="2018-11-17T09:30:47.0Z"
Version="6.49" xmlns="http://www.sabre.com/eps/schemas"
        >
        <ProfileInfo>
        <Profile CreateDateTime="2018-02-18T12:30:55.037"
                        UpdateDateTime="2018-02-26T09:26:12.211"
           PrimaryLanguageIDCode="EN-US">
                <TPA_Identity UniqueID="4259" ProfileTypeCode="TVL"
        ClientContextCode="EXT" ClientCode="AS" ProfileStatusCode="AC"  DomainID=“XX”>
                <Login LoginID="abc@example.com"
                                        PasswordHash="gra8EoFe8RBh+iBMY+ =" />
                </TPA_Identity>
            <Traveler>
                    <!—NOTE: the contents of the Profile sub tree are the same as can be seen in the
                        Sabre_OTA_ProfileCreateRQ traveler section above. -->
        </Traveler>
        </Profile>
        </ProfileInfo>
</Sabre_OTA_ProfileUpdateRQ>
```

Note In order to successfully update the profile (full Update), it is necessary to use proper values in the *ProfileUpdateRQ*. The following attributes are crucial: *CreateDateTime, UpdateDateTime, UniqueID, ProfileTypeCode*, *ClientCode*, *ClientContextCode* and *DomainID*. The *UniqueID, ProfileTypeCode, ClientCode, ClientContextCode* and *DomainID* identify a specific profile, whereas the *UpdateDateTime* – a specific historical snapshot of a profile.

Within the Full Update process all profile data stored in the system is replaced with the data from the *ProfileUpdateRQ* request. It does not concern CreateDateTime attribute and it does not remove Login element if it is already in profile but is not included in the full update.

6.3 How to do a full update of profile using <IgnoreSubjectArea>

The Sabre_OTA_ProfileUpdateRQ has the ability to update only selected subject areas and do not overwrite those which are not included or which are included but are listed in IgnoreSubjectArea in ProfileUpdate request. In order to do so, a user in full ProfileUpdate request must include the subject area that should be updated while those subject areas which must remain unchanged, need to be listed in **<IgnoreSubjectArea>** element.

Many subject area names that can be used within IgnoreSubjectArea. . Please refer to the schema files for a complete list of supported areas.

Update with <IgnoreSubjectArea> is most frequently used by customers, who store Credit Card data in *CI* but, for example due to PCI regulation in the country of origin, they don't store CC data in the 3rd party system integrated with *CI*. Those 3rd party systems are not able to retrieve full details of PaymentForm elements (by not owning BTFOPView or OSCCVIEW security attributes in the EPR account they use to access *CI* web services) and therefore should not update PaymentForm elements in profiles stored in *CI*.

As stated above in the full ProfileUpdate request, all profile elements must be included in the payload otherwise elements not included in the request will be removed from profile stored in *CI*. If user does not have credentials which will allow him to obtain credit card/travel bank details then during data retrieval "NOACCESS" is returned instead of card number. Therefore, user will not be able to reuse existing credit card details in the update request. In such a case customer may include <IgnoreSubjectArea> for PaymentForm element and may not include PaymentForm details in the update request at all. In such a case PaymentForm information stored in *CI* will remain unchanged.

Sample xml is available below:

```
<Sabre_OTA_ProfileUpdateRQ Target="Production" TimeStamp="2018-10-03T09:22:25.987Z"
Version="6.49" xmlns="http://www.sabre.com/eps/schemas"
        >
      <ProfileInfo>
        <Profile CreateDateTime="2013-10-03T09:22:25.987Z" UpdateDateTime="2013-10-
03T09:22:25.987Z">
          <TPA_Identity ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
ProfileTypeCode="TVL" UniqueID="29843911"/>
          <Traveler>
            <Customer BirthDate="2013-10-03">
              <Email EmailTypeCode="BUS" EmailUsageCode="RPL" PurposeCode="ALL"
EmailAddress="ABC@EXAMPLE.COM" EmailRemark="EmailSmith" DisplaySequenceNo="1"
```

```
OrderSequenceNo="1" LanguageIDCode="EN" FormatTypeCode="BOTH"/>
            </Customer>
          </Traveler>
          <IgnoreSubjectArea>
            <SubjectAreaName>PaymentForm</SubjectAreaName>
          </IgnoreSubjectArea>
        </Profile>
      </ProfileInfo>
    </Sabre_OTA_ProfileUpdateRQ>
```

6.4 How to partially update a profile

In case of a partial update to a traveler there are three possibilities. The first is related to **modifying** an element or elements, which are indicated. Then, in the *ProfileUpdateRQ*, three important sections should be used: **<ModifySubtree>**, **<MatchSubtree>** and **<NewSubtree>**, which are responsible for finding the specific position in the xml tree, finding the specific element at the correct position and changing it to the new value.

A sample PartialUpdate using “modify functionality” is described below:

```
<Sabre_OTA_ProfileUpdateRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z"
Version="6.49"xmlns="http://www.sabre.com/eps/schemas"
        >
        <ProfileInfo>
<PartialUpdates>
        <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="12345"
DomainID="XX"/>
                <Modify>
                        <ModifySubtree child="/Profile/Traveler/TPA_Extensions/SSR"
AddIfNotFound="Y">
                                <MatchSubtree>
                  <Traveler>

                        <SSR ServiceTypeCode="BU" SSRCode="Stru" Text="SSR data2"/>

                </Traveler>
          </MatchSubtree>
        <NewSubtree>
                <Traveler>
                        <SSR ServiceTypeCode="BU" SSRCode="Strx" Text="SSR dataNew"/>

                </Traveler>
        </NewSubtree>
                   </ModifySubtree>
                </Modify>
</PartialUpdates>
        </ProfileInfo>
</Sabre_OTA_ProfileUpdateRQ>
```

In the **<TPA_Identity**> section, the user identifies which Profile will be updated. Then the user chooses which action will be performed and, in this particular case, the **<Modify>** element must be used. This element contains a child attribute <ModifySubtree>, which contains the XPath to specify the element which is supposed to be modified. Within the **<Modify>** the user must specify which one should it be by setting the **<MatchSubtree>** section with the element to be modified. Whereas in the **<NewSubtree>** section, the user must define a new element to replace the old one. The *CI* system will search for the old element in the database and if the search is successful, the element is replaced. If the *CI* system cannot find the old element or too little details were provided to uniquely identify an element, then an appropriate error message will be returned.

Note When performing the **<Modify>** action it is also possible to add a flag *AddIfNotFound* as an attribute in the **<ModifySubtree>**. Using this flag will cause the system to add a new element if data indicated in the ProfileUpdateRQ is not stored in the current version of the profile.

The second capability of partial update is related to an **addition** of a new element. In this scenario, in the *PartialUpdateRQ*, under <Add> element, the **<AddSubtree>** section must be used. This section consists of one attribute and one sub-section. The attribute (which is called 'Child') holds the XPath of the element which is supposed to be added. In the sub-section, which is called 'NewSubtree', the user must define a new element. If the element does not violate any of the *CI* internal constraints, it will be added to the *CI* database. Otherwise an appropriate error message will be returned. A sample **PartialUpdateRQ** is outlined below:

```
<Sabre_OTA_ProfileUpdateRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"
        >
 <ProfileInfo>
        <PartialUpdates>
                <TPA_Identity UniqueID="1758" ProfileTypeCode="TVL"
            ClientCode="AS" ClientContextCode="EXT" DomainID="XX" />
                        <Add>
                         <AddSubtree child="Profile/Traveler/Customer/Document">
                                <NewSubtree>
                                        <Traveler>
                                         <Document DocIssueLocation="Dallas"
                                                DocID="1234567" DocTypeCode="AW"
                                                BirthDate="1977-09-13"
                                                EffectiveDate="1977-08-13"
                                                ExpireDate="1997-08-13"

                                                DocIssueCountryCode="US"
                                                BirthPlace="Austin"
                                                BirthCountryCode="US"
                                                DocHolderNationalityCode="UK">
                                <DocHolderName>Adarsh K Gosu</DocHolderName>
                                          </Document>
                                        </Traveler>
                                 </NewSubtree>
                           </AddSubtree>
                        </Add>
        </PartialUpdates>
 </ProfileInfo>
```

</Sabre_OTA_ProfileUpdateRQ>

The last option that can be used is related to **deletion** of a specific element. In this scenario, under <Delete> element, the user specifies the **<DeleteSubtree>** section, which consists of the 'Child' attribute, and contains the XPath of the element to be removed, and the **<MatchSubtree>** sub-section, which defines the element to be deleted. An example **PartialUpdateRQ**, which covers this scenario, is shown below:

```
<Sabre_OTA_ProfileUpdateRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"
        >
 <ProfileInfo>
        <PartialUpdates>
          <TPA_Identity UniqueID="1758" ProfileTypeCode="TVL"
                        ClientCode="AS" ClientContextCode="EXT"
                        DomainID="XX"
                <Delete>
                 <DeleteSubtree child="Profile/Traveler/Customer/Document">
                        <MatchSubtree>
                                <Traveler>
                                 <Document DocIssueLocation="Dallas"
                                        DocID="1234567" DocTypeCode="AW"
                                        BirthDate="1967-08-13" EffectiveDate="1967-
                                        08-13" ExpireDate="1967-08-13"
                                        DocIssueCountryCode="US" BirthPlace="Austin"
                                        BirthCountryCode="US"
                                         DocHolderNationalityCode="UK">
                         <DocHolderName>Adarsh K Gosu</DocHolderName>
                                  </Document>
                                </Traveler>
                        </MatchSubtree>
                 </DeleteSubtree>
                </Delete>
          </PartialUpdates>
   </ProfileInfo>
</Sabre_OTA_ProfileUpdateRQ>
```

Note Partial Updates are supported for most elements in the Traveler (TVL) profile type and for a couple of elements in the Corporate and Agency profile types.

6.5 Sample XML Update Successful Response

A successful update results in the following response. Note the **<Success>** element:

```
<Sabre_OTA_ProfileUpdateRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-12-07T08:40:05.999Z" Version="6.49" CreateDateTime="2018-12-
07T08:27:05.608Z"
UpdateDateTime="2018-12-07T08:40:05.925Z">
  <ResponseMessage>
    <Success />
```

```
  </ResponseMessage>
  <Profile ClientCode="AS" ClientContextCode="EXT"  UniqueID="30831022"
ProfileTypeCode="TVL"
  ProfileNameModifyIndicator="Y" DomainID="XX"  ProfileStatusCode="AC">
  </Profile>
</Sabre_OTA_ProfileUpdateRS>
```

6.6 Sample XML Update Error Response

When, for some reason, a profile could not be either fully or partially updated, an error message is returned. Each error message contains an Errors section with an Error Code and a short description of the problem which was encountered during the profile update process. For the *CI* Update Service, each Error message is prefixed with **"U::".** A sample error message is shown below:

```
<Sabre_OTA_ProfileUpdateRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-12-07T08:42:49.861Z" Version="6.49">
  <ResponseMessage>
    <Errors>
      <ErrorMessage ErrorCode="302">
      U::NCERT502-2T20131007084249SU::Invalid Last Update Time
      Stamp. Profile probably has been updated by another client</ErrorMessage>
    </Errors>
  </ResponseMessage>
  <Profile ClientCode="AS" ClientContextCode="EXT" UniqueID="30831022"
ProfileTypeCode="TVL"
  ProfileNameModifyIndicator="Y" DomainID="XX" ProfileStatusCode="AC">
  </Profile>
</Sabre_OTA_ProfileUpdateRS>
```

Search Service 7

7.1 Searching for Profiles

Profile search functionality allows a user to specify the search criteria and to find the matching profiles in the *CI* database.

From the search functionality perspective, the most crucial attributes specified in the ProfileSearchRQ are: *SearchOperationType, ProfileNameOnly, ExcludeDeletedProfiles* and *IncludeAdminProfiles*. The first defines the logical operator between search criteria (there is only one operator: "AND"– if not provided, it will be automatically set to 'AND'). ProfileNameOnly defines which data of the matching profiles is actually retrieved from *CI* database. *ExcludeDeletedProfiles* and *IncludeAdminProfiles* additionally defines which profiles are actually retrieved from the *CI* database. When setting ExcludeDeletedProfiles="Y" then profiles with ProfileStatusCode="DL" are not included in search response, when setting IncludeAdminProfiles ="Y" then TVL profiles with subtype "CAD" are included. Along with these attributes, a user can specify two extra ones: *PageNumber* and *ReturnCount*. These two attributes are used to do a repetitive search. Attribute *CountAll* if set to "Y" will provide actual number of profiles which matches search criteria.

Note For all searches, the maximum result count is set to 250. In the absence of a specified count number, the default value is set to return 25. In case of *ProfileNameOnly='Y'* the default can be set to maximum 50.

Below example shows how the search criteria are grouped in the following sub-sections:

- Basic Search Criteria (ProfileTypeCode, DomainID)
- TPA_Identity Criteria (ClientCode, ClientContextCode, ProfileName, LoginID)
- PersonName (GivenName, Surname)
- Email (EmailAddress)
- Telephone (AnyPhoneEntry, FullPhoneNumber, ParsedPhoneNumber)
- AssociatedProfiles (AssocProfileType, AssocProfileName)
- Document (DocumentID, DocumentTypeCode, DocIssueCountryCode)

Note Please refer to Sabre_OTA_ProfileSearchRQ schema available on Sabre Dev Studio for latest set of search criteria.

The Basic Search Criteria must be specified in the incoming ProfileSearchRQ.

The *ProfileTypeCode* can be set either to the specific profile type (in case of traveler profiles it should be set to 'TVL') or to 'ALL'. In the latter case, the search for profiles will consider each and every profile type ('TVL', 'AGY', 'CRP', 'AGT,). The *DomainID* must be set to a specific value which is a 2-character IATA airline code.

If the *ProfileNameOnly* is set to *true (Y)*, in the ProfileSearchRS, the user will get only *TPA_Identity* elements of the matching profiles. Otherwise, the user will get the *TPA_Identity* sections along with other profile data. If there is more than one profile in ProfileSearchRS, then profile data is limited.

In the *TPA_Identity* Section there are two required attributes, which are the *ClientCode* (which is "AS" that stands for *Airline Solutions* clients) and the *ClientContextCode (*which identifies the calling system. Sabre *Airline Solutions* customers should use the *ClientContextCode="EXT")*. The rest of

search criteria attributes in this elementare optional, however, if any other qualifiers are added, they will narrow the results set.

7.2 Sample XML Search Request

```
<Sabre_OTA_ProfileSearchRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"
        >
   <ProfileSearchCriteria ProfileNameOnly="Y" SearchOperationType="AND">
      <TPA_Identity DomainID="XX" ClientCode="AS" ClientContextCode="XX"
         ProfileTypeCode="TVL" />
      <Traveler Surname="Adam" GivenName="Smith"/>
      <Email EmailAddress="Adam.smith*"/>
   </ProfileSearchCriteria>
</Sabre_OTA_ProfileSearchRQ>
```

In the example above, the *ProfileNameOnly* attribute has been set to ‘Y’, which means that only the *TPA_Identity* element of matching profiles will be returned.

Apart from the Basic Search Criteria, which have been described earlier, each search criteria can contain a ‘*’ sign to serve as a “wildcard”:

Search Criteria = ‘*’ – all values of Search Criteria satisfy this condition

Search Criteria = ‘ABC’ – this is an exact match

Search Criteria = ‘AB*’ – this condition is satisfied by values that start from ‘AB’, which is followed by zero or more characters: ABXXX, ABC123, ABC, etc.

Note The “*” as wildcard is supported only at the end of the string. Itis not supported as a prefix (i.e. =’*AB’) to search for any values that are followed by letter AB.

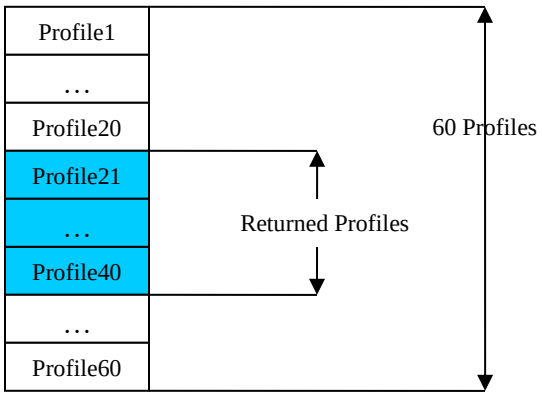
By defining the SortPreference sub-element, a user can specify the order profiles are returned in the response message which matches the search criteria. Profiles can be sorted in three ways:

- by ProfileName
- by their creation dates
- by their UniqueID

7.3 Repetitive search and a sample XML ProfileSearch request

Apart from the *SearchOperationTyp*e, *ExcludeDeletedProfiles* and *IncludeAdminProfiles* a user can specify the following two attributes: *PageNumber* and *ReturnCount*. These two attributes are used to do the Repetitive Search. This extra functionality has been added in order to provide a mechanism to retrieve only a sub-set of matching profiles (the exact number of matching profiles the *CI* search engine is supposed to return is determined by the *ReturnCount*) starting from a specific profile, which is determined by the PageNumber. Basically, this functionality covers the “Give me the next (n) profiles” scenario.

To illustrate how this functionality actually works, lets assume there are 60 profiles that match the search criteria and the user sets the ReturnCount to 20, and the PageNumber to 2 and runs ProfileSearchRQ.



The profiles returned in ProfileSearchRS are the subset that is highlighted in blue on the graph aove. Now, in order to get next 20 profiles, the user must increase the PageNumber while keeping other parameters as they were.

Note Please note that maximum *ReturnCount* value is 250 per page. When ReturnCount value is not defined then by default- 25 profiles will be returned.

A sample Search request that includes the Repetitive Search is shown below:

```
<Sabre_OTA_ProfileSearchRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"
        >
    <ProfileSearchCriteria PageNumber="2" ReturnCount="20" >
        <TPA_Identity DomainID="XX" ClientCode="AS" ClientContextCode="EXT "
ProfileTypeCode="TVL"/>
        <Traveler Surname="Bear" GivenName="Vernon"/>
        <Email EmailAddress="Bear.vernon*" />
    </ProfileSearchCriteria>
</Sabre_OTA_ProfileSearchRQ>
```

7.4 Sample ProfileSearch RQ/RS with CountAll attribute

User is also able to obtain number of all profiles which match defined search criteria by including CountAll=”Y” attribute in a ProfileSearchRQ. Once the CountAll=”Y’ is included in a SearchRQ payload, then in ProfileSearchRS TotalCount attribute contains actual number of profiles that match the search criteria given and additionally matching search criteria is also included in ProfileSearchRS. CountAll attribute has an effect only for the first page.

Here is a sample ProfileSearch request message where CountAll =”Y”.

```
<Sabre_OTA_ProfileSearchRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"
```

```
        >
  <ProfileSearchCriteria SearchOperationType="AND" ExcludeDeletedProfiles="Y"
CountAll="Y">
    <TPA_Identity ClientContextCode="EXT" ClientCode="AS" ProfileTypeCode="TVL"
DomainID="XX" />
    <Traveler Surname="Test*"/>
  </ProfileSearchCriteria>
</Sabre_OTA_ProfileSearchRQ>
```

Below is a sample ProfileSearch response for the request:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49">
  <ResponseMessage>
    <Success />
  </ResponseMessage>
  <ProfileInfo>
    <Message>Count: 25</Message>
    <ProfileList NumReturned="25" HaveMore="Y" TotalCount="837">
      <Profile>
                                    <!—All information about travelers-->

      </Profile>
    </ProfileList>
  </ProfileInfo>
</Sabre_OTA_ProfileSearchRS>
```

7.5 Sample ProfileSearchRS to the ProfileSearchRQ havingProfileNameOnly attribute set to “Y”

Hereunder is a sample ProfileSearch response message where ProfileSearch request has the ProfileNameOnly set to ‘Y’:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2018-12-20T16:52:23.561Z" Version="6.49">
        <ResponseMessage>
                <Success />
        </ResponseMessage>
        <ProfileInfo>
                <Message>Count: 2</Message>
                <ProfileList NumReturned="2" HaveMore="N" PageNumber="1" >
                    <Profile>
                      <TPA_Identity ClientCode="AS" ClientContextCode="CI"
                      UniqueID="12345" ProfileTypeCode="TVL"  DomainID="XX"
                      ProfileStatusCode="AC">
                        <Login LoginID="12345" />
                      </TPA_Identity>
                    </Profile>

                    <Profile>
                      <TPA_Identity ClientCode="AS" ClientContextCode="CI"
```

```
                    UniqueID="23456" ProfileTypeCode="TVL"  DomainID="XX"
                    ProfileStatusCode="AC">
                      <Login LoginID="23456" />
                    </TPA_Identity>
                  </Profile>
              </ProfileList>
        </ProfileInfo>
</Sabre_OTA_ProfileSearchRS>
```

The main part of the response message is the ProfileInfo section. The following information is found in this section:

- The number of profiles that matched the search criteria.
- The *TPA_Identity* sections of all the profiles that matched the search criteria.

7.6 Sample ProfileSearchRS single profile in response

If there is only one profile which matches the search criteria, the response message is as follows:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2018-12-20T16:52:23.561Z" Version="6.49">
        <ResponseMessage>
                <Success />
        </ResponseMessage>
        <ProfileInfo>

                                <Profile CreateDateTime="2018-03-03T12:08:03.19"
UpdateDateTime="2018-03-03T12:08:03.19" PrimaryLanguageIDCode="EN-US">
                        <TPA_Identity UniqueID="4351" ProfileTypeCode="TVL"
ClientCode="AS" ClientContextCode="XX" DomainID="XX" ProfileStatusCode="AC">
                        </TPA_Identity>
                        <Traveler>
                                <!_All information about traveler-->
                        </Traveler>
                </Profile>

        </ProfileInfo>
</Sabre_OTA_ProfileSearchRS>
```

When only one profile matches the search criteria, all profile data is returned (similar to a ProfileReadRQ) from the *CI* database (not only the TPA_Identity section).

7.7 Sample ProfileSearchRS multiple profiles in response

If more than one profile matches the search criteria, the following information is retrieved from the *CI* database for each matching profile:

- TPA_Identity
- PersonName

- Telephone
- Email
- Address
- AssociatedProfiles
- Customer Reference Info
- BirthDate
- CustLoyalty
- Document

Note Please refer to **Sabre_OTA_ProfileSearchRS** posted on Sabre Dev Studio for more information.

A sample Response message is as follows:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2018-12-20T16:52:23.561Z" Version="6.49">
        <ResponseMessage>
                <Success />
        </ResponseMessage>
        <ProfileInfo>
                <Message>Count: 2</Message>
                <ProfileList NumReturned="2" HaveMore="N" PageNumber="1">
                        <Profile>
                                <TPA_Identity ProfileID="4277" ProfileTypeCode="TVL"
ClientCode="AS" " DomainID="XX">
                                </TPA_Identity>
                                <Traveler>
                                        <!—-Profile information: PersonName, Telephone, Email,
                                        Address, Associated Profiles Customer Reference Info-->
                                </Traveler>
                        </Profile>
                                                        <Profile>
                                <TPA_Identity UniqueID="4278" ProfileTypeCode="TVL"
ClientCode="AS" ClientContextCode="EXT" DomainID="XX">
                                </TPA_Identity>
                                <Traveler>
                                        <!—-Profile information: PersonName, Telephone, Email,
                                        Address, Associated Profiles Customer Reference Info-->
                                </Traveler>
                        </Profile>
                </ProfileList>
        </ProfileInfo>
</Sabre_OTA_ProfileSearchRS>
```

As in the example above, along with TPA_Identity sections some other data is retrieved for each profile.

7.8 Sample ProfileSearchRS with HaveMore attribute set to “Y”

The maximum number of profiles that can be retrieved from the *CI* database is set to 250 per page. By default, the number of profiles returned is set to 25, but it can be changed up to the maximum value in ProfileSearch request by setting a value of ReturnCount attribute to “250”. If the number of profiles matching the search criteria exceeds requested number of returned profiles, then HaveMore indicator is set to “Y” and the following response message is returned:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2018-12-20T16:52:23.561Z" Version="6.49">
        <ResponseMessage>
                <Success />
        </ResponseMessage>
        <ProfileInfo>
                <Message>Count: 20</Message>
                <ProfileList NumReturned="20" HaveMore="Y" PageNumber="1">
                        <Profile>
                                <TPA_Identity ProfileID="4277" ProfileTypeCode="TVL"
ClientCode="EXT"  ClientContextCode="XX"  DomainID="XX">
                                </TPA_Identity>
                                <Traveler>
                                        <!—-Profile information: PersonName, Telephone, Email,
                                        Address, Associated Profiles Customer Reference Info-->
                                </Traveler>
                        </Profile>


                        <Profile>
                                <TPA_Identity ProfileID="4278" ProfileTypeCode="TVL"
ClientCode="EXT" ClientContextCode="XX" DomainID="XX">
                                </TPA_Identity>
                                <Traveler>
                                        <!—-Profile information : PersonName, Telephone,
                                        Email, Address, Associated Profiles Customer Reference
                                        Info-->
                                </Traveler>
                        </Profile>

                        <-- Remaining profiles -->

                </ProfileList>
        </ProfileInfo>
</Sabre_OTA_ProfileSearchRS
```

7.9 Sample ProfileSearchRS with no results

If there are no profiles in the *CI* database that match the search criteria, then the following successful response message is returned:

```
<Sabre_OTA_ProfileSearchRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-12-20T17:10:29.841Z" Version="6.49">
```

```
  <ResponseMessage>
    <Success />
  </ResponseMessage>

  <ProfileInfo>
    <Message>No profiles are found which match your selection criteria</Message>
  </ProfileInfo>
</Sabre_OTA_ProfileSearchRS>
```

7.10 Sample ProfileSearchRS with error response

When a search operation cannot be performed for some reason, an error response message is returned. Each error message contains Errors element followed by an ErrorMessage element with an ErrorCode attribute and a short description of the problem that was encountered during the profile search process. A sample error message is shown below:

For the *CI* ProfileSearch service, each ErrorMessage value is prefixed with "**S::".** An example error message is shown below:

```
<Sabre_OTA_ProfileSearchRS
xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-12-20T17:10:29.841Z" Version="6.49">
    <ResponseMessage>

        <Errors>
          <ErrorMessage ErrorCode="338">
          S::NINT1T20181220171518SS::Invalid XML (not compliant with Schema): Attribute
    @Surname is not allowed on element Login. One or more validation errors were
    reported.</ErrorMessage>
        </Errors>
    </ResponseMessage>
</Sabre_OTA_ProfileSearchRS>
```

Delete / Restore Service 8

8.1 Deleting and Restoring Profiles

The ProfileDelete service is a “mark for delete” functionality that changes profile statuts to “DL” (ProfileStatus="DL"). It allows to mark a specified profile to be deleted, so that it can be removed from the *CI* database by purge job after the predetermined time. The user can specify only one profile in the ProfileDeleteRQ and include number of days after which the profile should be removed (PurgeDays="0"). After each purge job daily run, the PurgeDays value is decreased by 1. The purge job looks for all the “marked for delete” (ProfileStatus="DL") profiles that have the lowest PurgeDays value. Purge job has a limit of maximum profile count that can be removed in a single run. The profiles which satisfy these conditions and fit into daily limit will be permanently removed from the *CI* database. Purge job starts every day at midnight U.S. Central Time.

If the ProfileStatus is set to delete with a PurgeDate indicator set to 0, it is considered as a request for an immediate purge and can be restored using the ProfileDelete service with restore option before it is purged.

Note The profiles that were deleted with standard PurgeDays="0" are generally purged from CI date base up to 48h later than scheduled in the request.

Note Alliance (ALC) profiles do not fall under this process, those profiles are purged immediately after the ProfielDelete request is received.

8.2 Sample ProfileDelete request Delete option

A sample ProfileDelete request with minimum time to purge expectation (PurgeDays="0"):

```
<Sabre_OTA_ProfileDeleteRQ Target="Production" TimeStamp="2018-10-03T09:22:25.987Z"
Version="6.49" xmlns="http://www.sabre.com/eps/schemas"
        >
        <Delete>
                <Profile PurgeDays="0">
                        <TPA_Identity ClientCode="AS" ClientContextCode="EXT"
DomainID="XX" ProfileTypeCode="TVL" UniqueID="3144924754"/>
                </Profile>
        </Delete>
</Sabre_OTA_ProfileDeleteRQ>
```

The **<Profile>** sub-element holds the *PurgeDays* attribute, which is mandatory. The value of this attribute is used by the *CI* purge job. Inside the **<Profile>** there is a *TPA_Identity* section. This section specifies a profile to be deleted.

There is no limit for the number of days in PurgeDays attribute. It should not be less than 0. If it is greater than 0 then it means that profile is going to be scheduled to be purged in advance.For example by setting PurgeDays="30", the profile should be purged 30 days from the current date.

Note Profiles are not automatically deleted from *CI* system regardless of the activity on the profile or it’s creation date. Therefore **Sabre_OTA_ProfileDeleteRQ** must be used if any profile needs to be deleted.

8.3 Sample ProfileDeleteRS successful response

A sample successful ProfileDeleteRS is as follows:

```
<Sabre_OTA_ProfileDeleteRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2018-10-04T12:38:53.008Z" Version="6.49">
          <ResponseMessage>
            <Success />
          </ResponseMessage>
          <Delete>
            <Profile PurgeDays=”0”>
              <TPA_Identity ClientCode="AS" ClientContextCode="EXT"
              UniqueID="72995785973" ProfileTypeCode="TVL" DomainID="XX" />
            </Profile>
          </Delete>
        </Sabre_OTA_ProfileDeleteRS>
```

In case of successful responses, the <ResponseMessage> element holds the <Success/> sub-element.

8.4 Sample ProfileDeleteRS error response

If a profile cannot be marked for delete for some reason, an error message is returned. Each error message contains an Errors element with an ErrorMessage element contianint ErrorCode attribute and a short description of the problem which was encountered during the profile delete or restore process. A sample error message is shown below:

For the *CI* ProfileDelete service (delete and restore option), each ErrorMessage is prefixed with “**D::”**. An example error message is shown below:

```
<Sabre_OTA_ProfileDeleteRS xmlns="http://www.sabre.com/eps/schemas"
        TimeStamp="2013-10-04T12:33:41.142Z" Version="6.49">
          <ResponseMessage>
            <Errors>
              <ErrorMessage ErrorCode="298">
              D::NCERT502-1T20131004123341SD::No profile is found which  match your selection
criteria (UniqueID = 3144924754,  ClientCode = AS, ClientContextCode = CI, DomainID = XX,
ProfileTypeCode = TVL, LoginID = null)</ErrorMessage>
            </Errors>
          </ResponseMessage>
          <Delete>
            <Profile>
              <TPA_Identity ClientCode="AS" ClientContextCode="CI"
              UniqueID="3144924754" ProfileTypeCode="TVL" DomainID="XX" />
            </Profile>
          </Delete>
</Sabre_OTA_ProfileDeleteRS>
```

In the example above, the profile could not have been deleted, because it was not found in the *CI* database.

```
<Sabre_OTA_ProfileDeleteRS xmlns="http://www.sabre.com/eps/schemas"
```

```
TimeStamp="2018-12-21T10:59:58.617Z" Version="6.49">
  <ResponseMessage>
    <Errors>
      <ErrorMessage ErrorCode="409">D::NINT2T20181221105958SD::Cannot retrieve Profile from
      database</ErrorMessage>
    </Errors>
  </ResponseMessage>

  <Restore>
    <Profile>
      <TPA_Identity ClientCode="AS" ClientContextCode="CI"
      UniqueID="800089393" ProfileTypeCode="TVL" DomainID="XX" />
    </Profile>
  </Restore>
</Sabre_OTA_ProfileDeleteRS>
```

In the example above, the profile could not have been restored, because it was not found in the *CI* database.

8.5 SampleProfileDelete service Restore option

Within the ProfileDelete service there is also an option to **Restore** a deleted profile instead of deleting it. This action changes the ProfileStatus from “DL” to “AC” (Active status) for profiles that are still in status “DL” which means that they were not yet purged.

Example below shows a ProfileDelete RQ and RS with a <Restore> element option:

```
<Sabre_OTA_ProfileDeleteRQ
Target="Production" TimeStamp="2018-10-03T09:22:25.987Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
        <Restore>
                <Profile>
                        <TPA_Identity ClientCode="AS" ClientContextCode="EXT"
DomainID="XX" ProfileTypeCode="TVL" UniqueID="72995785973"/>
                </Profile>
        </Restore>
</Sabre_OTA_ProfileDeleteRQ>

<Sabre_OTA_ProfileDeleteRS xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2018-12-21T11:04:18.151Z" Version="6.49">
  <ResponseMessage>
    <Success />
  </ResponseMessage>

  <Restore>
    <Profile>
      <TPA_Identity ClientCode="AS" ClientContextCode="CI"
      UniqueID="72995785973" ProfileTypeCode="TVL" DomainID="XX" />
    </Profile>
  </Restore>
</Sabre_OTA_ProfileDeleteRS>
```

Profile Index

9

Although the **Profile Index** (PI) is not a specific *CI* system feature, it is a Sabre host/TPF PNR function. Because of its relationship with the Profile functionality in *CI*, some basic information for this PNR field is described below.

Profile Index is a field available within a PNR that provides details of all Profiles copied into the PNR. A banner is visible within the PNR display as a reminder that Profile Index Data is present. The banner also provides the Sabre command that is available to display the data. Please find it below:

PROFILE INDEX DATA EXISTS *PI TO DISPLAY ALL

Each profile type copied into the PNR is represented by a Profile tag. Profile tags are either name associated or apply to all names in the PNR. A name associated tag applies to only one name in the PNR. The profile tags currently available are:

Profile Tag	Profile Type	Name Associated?
TAGENCY	AGENCY	NO
TAGENT	AGENT	NO
CORPID	CORPORATE	NO
TRAVELER	TRAVELER	YES

The elements of the Profile Index include the Profile tag and the Profile ID number. For a name-associated tag, the name, number, and traveler name are also shown.

9.1 PNR Profile Index display

Below are examples of how the Profile Index, which contains the Profile UniqueID, is displayed in the PNR. Display logic that formats the output exists for this move process as noted below.

***PI«**
PROFILE INDEX DATA
1.TRAVELER 1.1 JOHNSON/JOHN MR
6560015693
2.CORPID
123456789
3. TAGENCY
99002996
4.TAGENT
12000000423

- Any previous existing **PI** tags are copied into the Profile Index history in the PNR if the record was ended after the transaction.
- Multiple instances of the **Traveler Profile** tag can exist, but only one is allowed per name in the PNR.

Note The PI field contains only the Profile UniqueID (or IATA number for Agency profiles). It does not have a Domain or PCC information stored within the PNR.

Note With a PI field being part of a PNR other Sabre applications are able to associate Profiles with PNRs. Sabre applications may then use this information to display list of active PNRs for a particular profile.

Notification Service (PNS) 10

10.1 Profile Notification Service Overview

Intention of this chapter is to provide overview of the Profile Notification Service functionality and allow better understanding of all profile services. If customer would like to implement Notification Service please refer to dedicated Notification Service document called "**Sabre® Profiles Notification Service Developer's Guide to Getting Started".** In order to get a copy, please contact Sabre Customer Insight team.

While this is a standard feature of Customer Insight, there is an implementation fee to activate Profile Notification Service. Please contact your Account Director if you would like additional information.

10.2 About Profiles Notification Service

With Profile Notification Service, subscribing entities receive notifications whenever events to which they subscribe occur. For example, when a profile is created or modified, PNS sends a brief alert message that includes details about the new profile or changes to an existing profile (profile ID number, action, timestamp, and subject area) to the subscribing application.

The events available in this release of the product are profiles:

- Create
- Update
- Delete
- Restore

Profiles Notification Service is based on the public draft release of the Web Services Eventing Specification of August 2004. All notifications are OTA XML compliant.

Profile Notification Service sends notifications to the remote URL of subscriber called further in this document as the "event sync". Notifications are sent in nearly real-time after event occurs. Subscriber is expected to reply back with a response to a notification request confirming that notification was correctly delivered.

When an event that matches parameters defined by subscriber is triggered, then *event sync* listener application receives a notification in structured XML format via a protocol defined in a subscription questionnaire. If the event does not match the specified parameters, then no event gets triggered.

The notifications are delivered as XML messages. Sabre Profile Notification Service uses Sabre Supplier Side Gateway (SSG) to send notifications. Available communication protocols are listed with other technical questions in the SSG Questionnaire document. Tto get a copy, please contact Sabre Customer Insight team.

High Reliability

Profile Notification Service is designed to deliver messages with high reliability. The goal for all components that comprise the infrastructure is to be available 24x7, with the exception of scheduled maintenance. A window for maintenance activities will be established at a regular time and communicated to subscribing entities. When the need for maintenance arises, *Profile Notification Service* will be unavailable during this maintenance window or on as needed basis.

Fast Delivery

The Profiles Notification Service infrastructure provides nearly real-time delivery of notifications. The notification is pushed when an event occurs.

Throttling of Notification Delivery per Endpoint

During configuration process the subscriber defines throttling limits for subscription. Profiles Notification Service guarantees to not exceed the agreed throttling limit to not flood the *event sync* application. “Sabre® Profiles Notification Service- Developer’s Guide to Getting Started” document describes how to define those parameters.

Delivery Retry

If *Profile* Notification Service cannot deliver the message because of communication issues, it retries delivery after the communication is established.

If a notification is delivered, but the business exception occurs (for example *event sync* application replies with an error because of some internal process fails), the Profile Notification Service will retry to send the notification up to 5 times. If all 5 attempts to deliver the notification fail, then the event in CI system will be marked as failed. Report of such case is going to be included in an email This e-mail will contain information that sending the notification for given UniqueID was not possible.

Below is an example of such e-mail

From: CustomerInsight@sabre.com <CustomerInsight@sabre.com>
Sent: Sunday, October 28, 2018 6:09 PM
To: Profiles-prod-CI-errors Profiles-prod-CI-errors@sabre.com;(...)
Subject: Profile Notification - Notification failed after re-try -Domain: XX

Profile Notification -
A communication exception has ocurred and the system has been unable to deliver notification after re-try attemps. ***2018-10-28 17:09:02.181 UTC*** *for Domain: XX*
Unique ID: 00012233445
The notication is now suspended.
Notificiations for profiles stored in queue will resume after communication has been restored.

Sabre Customer Insight - Customer Profiles

Note: This email was sent from a notification-only address that cannot accept incoming email. Please do not reply to this message. If you need additional assistance please contact "Profiles-Customer-Support@sabre.com"

Profile Merge Service 11

11.1 Merging Profiles

CI web service clients are able to merge profiles using the **Sabre_OTA_ProfileMergeRQ**. Profile Merge is an orchestrated web service that is designed to handle the merging of duplicate profiles and update the Profile Index reference in active PNRs. Synchronization of merged profiles needs to be first enabled in Customer Insight to make it work.

The action code that should be used by Airline Solutions customers when calling this service is: **EPS_AS_EXT_ProfileMergeRQ.**

This service does not identify duplicate profiles. After a customer has identified the existence of one or more duplicated profiles for the same individual, then those profiles can be merged. A trace of that fact is stored in a merged profile for reference. Any PNRs that have been indexed with a profile that is being merged as a duplicate, can be updated to reference the profile ID of a "Master profile" and remove the PI profile index of the previous profile that is now identified as "Duplicate".

11.2 How to Merge a Profile

- On each transaction (service call) the profile Merge service allows only for a one to one relationship between profile identified as "Master" and one profile identified as "Duplicate" to be merged. However, the process can be called as many times as needed if more duplicate records need to be merged into one profile (considered as Master).
- When a duplicate profile is merged it gets disabled for access (read, search, update) by assigning a Merged status code (MG). In addition duplicated profile is updated with <AssociatedProfile> section with reference to Master profile UniqueID number duplicated profile got merged into. This way the profile record is retained in the system for reference with information of Master profile.
- The Master profile is updated with "association" reference in the TPA_Extension area indicating a "Profile relationship" of merged code (MG) as a type of association:

  …<TPA_Extensions>
  <AssociatedProfiles AssocUniqueID="121" AssocProfileTypeCode="TVL" DomainID="XX" ClientCode="AS" ClientContextCode="GLB" **ProfileRelationTypeCode="MG"** ProfileRelationStatusCode="AC" CreditBankIndicator="N" AssocFiltersInd="N"/>
  <NumberOfAssocProfiles Traveler="1"/>
  </TPA_Extensions>…

- No other data is added, modified or combined in the profiles during the Merge process.
- The ProfileMerge service request will generate a subsequent request to Trip Data Services system where it is expected to update active PNRs. This update should be done in real-time and apply only to those PNRs referencing to the duplicate profile, the process updates Profile Index field referencing the duplicate profile with the Master profile ID.
- If for any reason, the PNR update does not return a successful response, the Profile merge will be stopped and no changes will be made to the profiles, therefore allowing user to re-try the ProfileMerge at a later time.

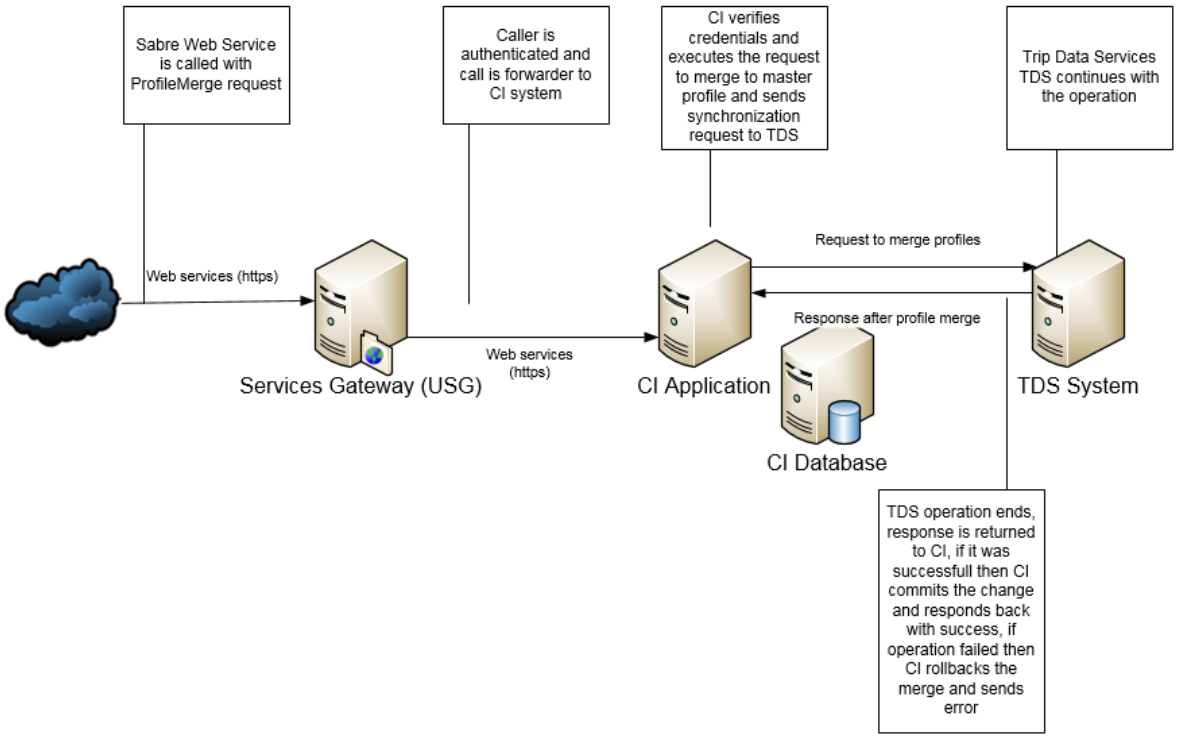
- The profile merge can only be executed for profiles that are in Active status (AC). ProfileMerge cannot be executed for a profile that is already in Merge (MG) ProfileStatusCode.

….<Profile CreateDateTime="2011-08-24T14:51:56.55Z" UpdateDateTime="2011-08-25T22:01:18.492Z" PrimaryLanguageIDCode="EN">
<TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="122" ProfileTypeCode="TVL" ProfileNameModifyIndicator="Y" DomainID="XX" ProfileStatusCode="**MG**"/>
<Traveler>
<Customer>
<PersonName LanguageIDCode="EN-US">
<GivenName>PATY</GivenName>
<SurName>ENRIQUEZ</SurName>…

11.3 System diagram illustrating flow for Merge process

The diagram below presents an illustration of the merge process where the ProfileMerge web service is invoked by consuming application after identifying duplicate profiles to be merged. The web service request is processed via USG (Universal Service Gateway) interface reaching out to CI/PPP system to conduct the merging process for profiles. CI system also connects to TDS (Trip Data Services) to conduct the corresponding active PNR Profile Index updates removing previous index and replacing it with newly identified "master" profile identifier. This allows cross referencing of trips transactions to the corresponding profile.

ProfileMergeRQ/@ActivePNRListIndicator flag allows to control the response of the service and if set to "Y" (Yes), response includes a list of active PNRs that have been updated.



11.4 Sample XML Profile Merge Request

Example 1: ActivePNRListIndicator=”N” (do not return a list of updated active PNRs) (If this attribute is not set then by default it has value: “N”).

```
<Sabre_OTA_ProfileMergeRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" ActivePNRListIndicator=”N”>
  <MasterProfile>
   <TPA_Identity
    ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
    ProfileTypeCode="TVL" UniqueID="1000000303"/>
<!—This section identifies the Profile that will be retained as Master and to which an association will
be added form the Merged/duplicate profile à
  </MasterProfile>
  <DuplicateProfile>
   <TPA_Identity
    ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
    ProfileTypeCode="TVL" UniqueID="1000000304"/>

  </DuplicateProfile>
 </Sabre_OTA_ProfileMergeRQ>
```

Response example:

```
<Sabre_OTA_ProfileMergeRS TimeSta”p="2011-04-05T22:19:11.”2Z" Versi”n="6.49"
xmlns="http://www.sabre.com/eps/schemas">
      <ResponseMessage>
        <Success/>
      </ResponseMessage>
      <ResponseData>
        <MasterProfile>
          <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="1000000303"
ProfileTypeCode="TVL" DomainID="XX"/>
        </MasterProfile>
        <MergedProfile>
          <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="1000000304"
ProfileTypeCode="TVL" DomainID="XX"/>
        </MergedProfile>
      </ResponseData>
    <Sabre_OTA_ProfileMergeRS>
```

Example 2: ActivePNRListIndicator=”Y” (return list of updated active PNRs)

```
<Sabre_OTA_ProfileMergeRQ
```

```
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" xmlns:xsi="http://www.w3.org/2018/XMLSchema-
instance" xsi:schemaLocation="http://www.sabre.com/eps/schemas
..\schemas\Sabre_OTA_ProfileMergeRQ.xsd" ActivePNRListIndicator="Y">
  <MasterProfile>
   <TPA_Identity
    ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
    ProfileTypeCode="TVL" UniqueID="1000000303"/>
<!—This section identifies the Profile that will be retained as Master and to which an association will
be added form the Merged/duplicate profile -->
  </MasterProfile>
  <DuplicateProfile>
   <TPA_Identity
    ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
    ProfileTypeCode="TVL" UniqueID="1000000304"/>
<!--This section identifies the Profile that will be set as Merged profile, de-activating access to it by
setting “MG” merged profile status.  When the customer is configured in TripDataServices to update
the Profile Index “PI field” in PNR during a ProfileMerge in CI, then this Profile ID will replaced in
PNR with the Profile ID as in MasterProfile element -->
  </DuplicateProfile>
 </Sabre_OTA_ProfileMergeRQ>

<Sabre_OTA_ProfileMergeRS TimeStamp="2011-04-05T22:19:11.02Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
      <ResponseMessage>
        <Success/>
      </ResponseMessage>
      <ResponseData>
        <MasterProfile>
          <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="1000000303"
ProfileTypeCode="TVL" DomainID="XX"/>
        </MasterProfile>
        <MergedProfile>
          <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="1000000304"
ProfileTypeCode="TVL" DomainID="XX"/>
         <ActivePNR>
                <PNR RecordLocator="RMT33W"/>
                <PNR RecordLocator="KZVGX5"/>
         </ActivePNR>
        </MergedProfile>
      </ResponseData>
     </ Sabre_OTA_ProfileMergeRS>
```

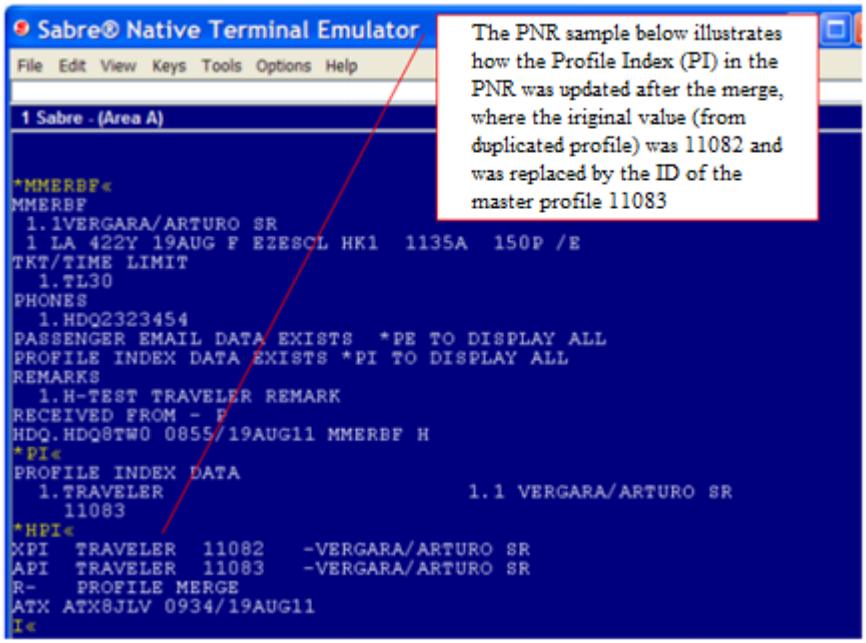
The following ProfileMergeRS example with WarningMessage illustrates a valid Profile merge action, when no PNRs were found to update. There were no active PNRs that had been Indexed (PI) with the Profile ID as provided in the MergedProfile/DuplicateProfile

```
 <Sabre_OTA_ProfileMergeRS TimeStamp="2011-08-25T18:53:10.397Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
      <ResponseMessage>
```

<Warnings>
<WarningMessage WarningCode="500524">No PNRs for given profile</WarningMessage>
</Warnings>
</ResponseMessage>
<ResponseData>
<MasterProfile>
<TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="1203456789" ProfileTypeCode="TVL" DomainID="XX"/>
</MasterProfile>
<MergedProfile>

<TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="12134599999" ProfileTypeCode="TVL" DomainID="XX"/>

</MergedProfile>
</ResponseData></Sabre_OTA_ProfileMergeRS>



11.5 Necessary system access and permissions

In case of AS customers security attributes are:

CustomerInsightExternalUser for external customers and applications

CustomerInsightUser for Sabre internal products.

Additionally when connecting with TDS to update PNR data, the user must have **PNRMoveUser**.

It is important to consider that to perform the Merge, the user must be passing the Session SecurityToken (ATHid) from the same partition/group where the PNRs belong. For example, to update airline's PNRs, the session must use the security token from the same partition/domain.

*All service calls assume the **SessionCreateRQ** has already returned the binary session token and conversation ID for the web client session being described .*

Profile Identity Info Read service 12

12.1 Profile Identity Info Read overview

The ProfileIdentityInfoRead (PIIR) was designed to provide passenger name section and frequent flyer tier extended information along with tier level mapping.

Customer is able to obtain tier level information for its own profiles as well as profiles set up for partner's airlines (when configured appropriately).

The PIIR service was optimized to provide limited profile information and tier level information. Along with its compact form it can be used for other purposes:

- name validation against internal set of conditions
- number validation against internal set of conditions
- access external system before providing final result
- access profiles of a separate partner domain that is hosted or non-hosted (but a part of alliance or following other agreements) in Sabre and has profile data in Customer Insight. Requestor (a system that is able to use PIIR) is able to obtain tier level information for its own profiles as well as profiles set up for partner's airline in Customer Insight.
- allows to obtain the "KnownTravelerNumber" and "KnownTravelerIndicator" for customers storing such information as part of the Transportation Security Administration expedited screening initiative.

While this is a standard feature of Customer Insight, there is an implementation fee to activate validations and cross domain access. Please contact your Account Director if you would like additional information.

12.2 Profile Identity Info Read Request Message

The request message of ProfileIdentityInfoRead service may contain up to 10 <Profile> elements for which profiles information are expected to be returned. In order to keep the service performance there is no plan to change this parameter.

Profile information might be requested by:

- providing a frequent flyer number in Profile/MembershipID element
- prividing a LoginID along with a DomainID profile belongs to
- providing a frequent flyer number in Profile/CustLoyalty/@MembershipID attribute, in this case name/number validations may be applied if previously configured

A sample request to retrieve two profiles is shown below:

```
<Sabre_OTA_ProfileIdentityInfoReadRQ Version="6.49"
xmlns="http://www.sabre.com/eps/schemas"  RequestTrackingID="1234sdf57t75756">
        <Profile DomainID="XX" ClientCode="AS">
                <MembershipID>7890123456</MembershipID>
```

```
        </Profile>
        <Profile DomainID="XY" ClientCode="AS" >
                <MembershipID>10082008</MembershipID>
        </Profile>
</Sabre_OTA_ProfileIdentityInfoReadRQ>
```

DomainID is a mandatory attribute, which designates the airline IATA code for which profile identity information is requested. MembershipID element is a frequent flyer card number used to identify a profile in order to retrieve it. In case this element is used then no validation of data between the ones in the request and the one in CI database will take place. ClientCode for all customers consuming CI web services must be set up as “AS”.

A sample request to retrieve a profile by LoginID is shown below:

```
<Sabre_OTA_ProfileIdentityInfoReadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" RequestTrackingID="1234sdf57t75756">
<Profile DomainID="XX" ClientCode="AS" >
                <Login LoginID="adam.smith@example.com"/>
        </Profile>
</Sabre_OTA_ProfileIdentityInfoReadRQ>
```

12.3 Profile Identity Info Read – Response Message

If the profiles are successfully retrieved, the response contains the passenger name, tier level and tier levels mapping, KnownTravelerNumber, Known Traveler Indicator if those exist in a profile.

A sample response is shown below:

```
<Sabre_OTA_ProfileIdentityInfoReadRS Version="6.49"
xmlns="http://www.sabre.com/eps/schemas" RequestTrackingID="1234sdf57t75756">
        <ResponseMessage>
                <Success/>
        </ResponseMessage>
        <Profile MembershipID="7890123456" DomainID="XX" ClientCode="AS"
ProfileStatusCode="AC" >
                <PersonName>
                        <GivenName>Anna</GivenName>
                        <SurName>Smith</SurName>
                </PersonName>
<Document DocID="123545687" DocTypeCode="KTID" OrderSequenceNo="1"/>
<Document DocID="58585858" DocTypeCode="KIND" OrderSequenceNo="2"/>
<CustLoyalty MembershipID="7890123456" VendorCode="XX" VendorTypeCode="AL">
<MembershipLevel>
                                <TierLevel TierLevelTypeCode="BN"
TierLevelValue="Platinum"/>
                                <TierLevel TierLevelTypeCode="2L" TierLevelValue="PL"/>
                                <TierLevel TierLevelTypeCode="3L" TierLevelValue="PLN"/>
                                <TierLevel TierLevelTypeCode="TI" TierLevelValue="04"/>
</MembershipLevel>
```

```
            </CustLoyalty>
        </Profile>
</Sabre_OTA_ProfileIdentityInfoReadRS>
```

12.4 Profile Identity Info Read response- Error Scenarios

If some (or all) of the requested profiles are not found, or if other problems occur during retrieval, an error response is returned. If an error occurs only for some profiles, but some are retrieved successfully, the response contains both error messages and profile content. A sample error message response is shown below:

```
<Sabre_OTA_ProfileIdentityInfoReadRS
xmlns="http://www.sabre.com/eps/schemas"
TimeStamp="2013-10-08T14:47:06.39Z" Version="6.49">
  <ResponseMessage>
    <Warnings>
      <WarningMessage WarningCode="20">Profile with
      MembershipID=7890123456 and DomainID=XX does not exist.</WarningMessage>
    </Warnings>
  </ResponseMessage>
  <Profile MembershipID="10082008" DomainID="XY" ClientCode="AS"
ProfileStatusCode="AC">
    <PersonName>
      <GivenName>Adam</GivenName>
      <SurName>Smith</SurName>
</PersonName>
 <CustLoyalty VendorTypeCode="AC" VendorCode="*A" MembershipID="10082008">
      <MembershipLevel>
        <TierLevel TierLevelTypeCode="TI" TierLevelValue="1" />
        <TierLevel TierLevelTypeCode="BN" TierLevelValue="Gold" />
        <TierLevel TierLevelTypeCode="3L" TierLevelValue="GLD" />
      </MembershipLevel>
    </CustLoyalty>
  </Profile>
</Sabre_OTA_ProfileIdentityInfoReadRS>
```

If profile does not contain a Frequent Flyer number in a Loyalty Program hosted by an airline then profile is not considered as a Frequent Flyer. In such a case, regardless of the request criteria, existing profile information is returned and a warning massage is added to the response. Please find example below.

```
<Sabre_OTA_ProfileIdentityInfoReadRQ TimeStamp="2018-11-30T12:00:00Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
        <Profile DomainID="XX" ClientCode="AS">
          <MembershipID>123123123</MembershipID>
        </Profile>
      </Sabre_OTA_ProfileIdentityInfoReadRQ>
```

```
<Sabre_OTA_ProfileIdentityInfoReadRS TimeStamp="2018-07-23T13:42:36.185Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
      <ResponseMessage>
        <Warnings>
<WarningMessage WarningCode="26">Profile MembershipID="123123123" is not valid Frequent
Flyer profile - CustLoyalty section does not exist for host airline.</WarningMessage>
        </Warnings>
      </ResponseMessage>
  <Profile MembershipID="123123123" DomainID="XX" ClientCode="AS"
ProfileStatusCode="AC">
    <PersonName LanguageIDCode="EN-US">
     <GivenName>Adam</GivenName>
     <SurName>Smith</SurName>
    </PersonName>
  </Profile>
</Sabre_OTA_ProfileIdentityInfoReadRS>
```

12.5 Profile Identity Info Read request - security

Customer is able to obtain profile information using ProfileIdentityInfoReadRQ for profiles set up in its own DomainID. It is also possible to obtain profile information via ProfileIdentityInfoReadRQ for partner airlines. Customer is also able to obtain tier level information for their partner airlines. However, access to other airline information must be granted by the corresponding partner airline and once approved, would need to be set up in *CI* configuration by the Customer Insight Delivery Team.

If a user tries to obtain profile information via ProfileIdentityInfoRead for an airline which is not considered a partner airline and access hasn't been granted by *CI* Delivery Team, the following warning will be included in the response:

```
<Sabre_OTA_ProfileIdentityInfoReadRS TimeStamp="2018-07-23T13:42:36.185Z" Version="6.49"
xmlns="http://www.sabre.com/eps/schemas">
      <ResponseMessage>
        <Warnings>
          <WarningMessage WarningCode="23">User is not authorized to read profiles in XX
domain.</WarningMessage>
        </Warnings>
      </ResponseMessage></Sabre_OTA_ProfileIdentityInfoReadRS>
```

Duplicate Check Service 13

13.1 Duplicate Check overview

When submitting a request to create a profile, the user can use the 'ProfileDuplicateCheck' flag to locate duplicate profiles or use the DuplicateCheck service. The DuplicateCheck service is used as a standalone service which can verify if duplicate profiles already exist in CI. Duplicated profiles are selected based on the duplicate pattern and level approved by a customer and set up in CI database during domain activation.

Currently, the following matching patterns are available:

- Name and Email address
- Name and Birthdate
- Name and Telephone Number

There are two matching levels to select from:

- Tight Match
- Loose Match

Note In order to use Duplicate Check service, this functionality must be first set up by the CI implementation team for the given domain.

Sample xml DuplicateCheckRQ

```
 <Sabre_OTA_ProfileDuplicateCheckRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" >
      <Profile>
        <TPA_Identity ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
ProfileTypeCode="TVL" UniqueID="*"/>
        <Traveler>
          <PersonName>
            <NamePrefix>MRS</NamePrefix>
            <GivenName>JEROLD</GivenName>
            <SurName>HOLBROOK</SurName>
          </PersonName>
          <Email EmailAddress="TEST_KJLZKH@EXAMPLE.COM"/>
        </Traveler>
      </Profile>
    </Sabre_OTA_ProfileDuplicateCheckRQ>
```

Sample xml DuplicateCheckRS

```
 <Sabre_OTA_ProfileDuplicateCheckRS TimeStamp="2013-10-24T08:09:08.745Z"
Target="Production" Version="6.49" MatchedProfileCount="1"
xmlns="http://www.sabre.com/eps/schemas">
       <ResponseMessage>
         <Success/>
       </ResponseMessage>
       <ProfileInfo>
         <Profile>
           <TPA_Identity ClientCode="AS" ClientContextCode="EXT" UniqueID="35369325"
ProfileTypeCode="TVL" ProfileNameModifyIndicator="Y" DomainID="XX"
ProfileStatusCode="AC"/>
           <Traveler>
             <PersonName>
               <NamePrefix>MRS</NamePrefix>
               <GivenName>JEROLD</GivenName>
               <SurName>HOLBROOK</SurName>
             </PersonName>
<Telephone>
       <ParsedPhoneNumber CountryCd="48" AreaCd="12"
       PhoneNumber="987987987" />
     </Telephone>
             <Email EmailAddress="TEST_KJLZKH@EXAMPLE.COM"/>
             <BirthDate>1938-10-22</BirthDate>
           </Traveler>
         </Profile>
       </ProfileInfo>
    </Sabre_OTA_ProfileDuplicateCheckRS>
```

Profile Batch Upload and Extract 14

Note Upload and extract profiles functionality is not directly available for customers to call via web service. The upload schema should rather be used to format the xml files and transmit them via SFTP for Customer Insight to process them off line.

14.1 Upload batch

The ProfileUploadRQ is a dedicated service for batch actions related to profiles.

The Sabre_OTA_ProfileUploadRQ schema allows for creation, modification and deletion profiles which are stored in CI database.

The schema contains two main elements:

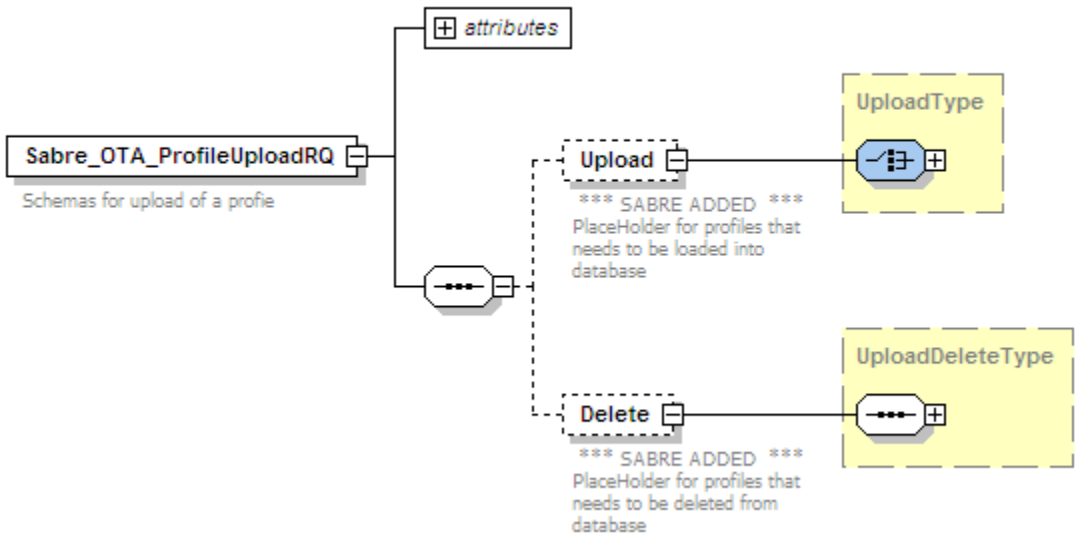
Upload

Delete

The **Upload** element is used to create or modify a profile in the Customer Insight database. During upload process CI application checks if record included in the upload file already exists in CI. If profile exists in CI, then profile data are modified, if profile does not exist then new profile is created in CI.

The **Delete** element is used to delete profile from CI database. If profile is sent in <Delete> section then CI set ProfileStatusCode=DL and then purged it from CI.

Screen shot below illustrate the <Sabre_OTA_ProfileUplaodRQ> schema. For more elements available in <Sabre_OTA_ProfileUploadRQ> check Sabre Dev Studio.



Sample XML Upload request

```
<Sabre_OTA_ProfileUploadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas"  >
<Upload>
<ProfileInfo >
<Profile CreateDateTime="2012-03-14T18:01:32.984Z" UpdateDateTime="2012-03-
14T18:01:32.984Z" PrimaryLanguageIDCode="EN">
   <TPA_Identity ClientCode="AS" ClientContextCode="UPL" UniqueID="866303992"
ProfileTypeCode="TVL" DomainID="XX" ProfileStatusCode="AC">
<Traveler>
                <!--Traveler information-->
</Traveler>
</Profile>
</ProfileInfo>
<ProfileInfo>
<Profile CreateDateTime="9999-12-31T00:00:00" UpdateDateTime="9999-12-31T00:00:00"
PrimaryLanguageIDCode="EN">
<TPA_Identity ClientContextCode="UPL" ClientCode="AS" UniqueID="325732186"
ProfileTypeCode="TVL" DomainID="XX" ProfileStatusCode="AC">
<Traveler>
                 <!--Traveler information-->
</Traveler>
</Profile>
</ProfileInfo>
</Upload>
 </Sabre_OTA_ProfileUploadRQ>
```

Please refer to the "Create Service" chapter for more details on actions triggered by these attributes.Following this section is base element **<Profile>** with the attributes: *CreateDateTime*, *UpdateDateTime* and *PrimaryLanguageIDCode*. *CreateDateTime*, *UpdateDateTime* are mandatory attributes however their values will be ignored. *PrimaryLanguageIDCode* is optional attribute. If the value of the *PrimaryLanguageIDCode* is not set, it'll be set to "EN-US" by default. The *PrimaryLanguageIDCode* can hold one of the values specified in the Dictionary Control Value XML files.

The following attributes: *UniqueID*, *ProfileType*, *ClientCode*, *ClientContextCode* and *DomainID* are required.

Further in a <Profile> element section customer details which should be stored in CI must be included.

Delete section in Sabre_OTA_ProfileUplaodRQ

The <Delete> section follows the <Upload> section. Profiles indicated in the <Delete> section will be set in ProfileStatusCode = DL and then if not restored, purged from Customer Insight database.

Both <Upload> and <Delete> section are optional in the <Sabre_OTA_ProfielUploadRQ>. Therefore customer may include <Upload> profiles, <Delete> profiles or both in one file.

Sample XML Delete request

```
<Delete>
<Profile PurgeDays="0">
<TPA_Identity  ClientCode="AS"  ClientContextCode="UPL"  UniqueID="123456789"
ProfileTypeCode="TVL"  DomainID="XX"/>
        </Profile>
        </Delete>
```

PurgeDays attribute is mandatory element and defines when profile should be deleted from CI database.

Note: Purge process starts 48h after it has been defined in the request. Please also note that purge process may take up to 48h depending on the load.

ClientCode, ClientContextCode, UniqueID , ProfileTypeCode, DomainID are mandatory elements and are used to defined profiles which should be deleted from CI.

14.2 Upload response

User is able to obtain upload result notification via email. In order to receive an upload notification email address for which notification should be delivered must be defined in CI database.

The information provided in the upload reports allows customer to identify failed record and reason of failure. Customer should fix the issue as soon as possible and resend the record to CI to keep CI and external system in synchronization.

The email notification contains the following content:

Dear Customer,

Please find report from upload process for CI_PRD_UP_XX_TVL_180610_2101_000.xml.errors.log

Total received: 14440
Success: 14440
Failed: 0

Thank you,

Customer Insight

Attachment contains details like Profile UniqueID and Error massage of failed profiles. Those attachments need to be encrypted and can be delivered in one of the two following options:

Option 1 - [DEFAULT] Encrypted attachment in e-mail – in this case the upload report file will still be sent via email, but the attachment will be encrypted. The file (.zip) will be compressed and secured with a one-time password which will be delivered in a separate email each time the report is sent.

Sample of email containg password:

Dear Customer,

This e-mail was generated in order to allow you to extract the upload report from an encrypted file that was sent in a separate e-mail with the same subject.

Please find the file name and password below.

Report file name: CI_PRD_UP_XX_TVL_180610_2101_000.xml.errors.log.sabre.zip
Password: bwNo9x0_irAb3f6uphl1O7FDHAa8exBh30iOygqtvI1jvj9.6heVbRarMVFj9hjW

Thank you,

Customer Insight

Option 2 - SSH File Transfer Protocol (SFTP) – for this option, the upload report containg name of the file which was processed, total recieved records, success and failed will still be sent via email but the attachment will be pushed to the SFTP server and catalogue pointed to by customer.

Note Example of upload report attachment content:

OrderSeqNo,Date,Operation,UniqueId,Result,Error

1,07/23/2013 01:20:22,DELETE,005014494,FAILURE,D::NPASHLP609-PRODBT20130723012022SD::No profile is found which match your selection criteria (UniqueID = 005014494 ClientCode = AS ClientContextCode = UPL DomainID = XX ProfileTypeCode = TVL LoginID = null)

14.3 Upload request – partial update

Customer Insight support partial changes via upload file. This kind of request is commonly used if customer wants to change only one/ few subject areas. Partial Update for CustLoyalty section which is updating Account Balance or MembershipLevelValue is commonly used type of request.

Sample UploadRQ xml – partial update

```
<Sabre_OTA_ProfileUploadRQ
TimeStamp="2018-11-17T09:30:47.0Z" Version="6.49" Target="Production"
xmlns="http://www.sabre.com/eps/schemas" >
<Upload>
<ProfileInfo>
<PartialUpdates>
<TPA_Identity ClientCode="AS" ClientContextCode="EXT" DomainID="XX"
ProfileTypeCode="TVL" UniqueID="222258105"/>
<Modify>
<ModifySubtree child="/Profile/Traveler/Customer/CustLoyalty">
<MatchSubtree>
```

<Traveler>
<CustLoyalty VendorTypeCode="AL" VendorCode="XX" MembershipID="222258105">
<MembershipLevel MembershipLevelTypeCode="TI" MembershipLevelValue="1"/>
<CustLoyaltyTotals AccountBalance="125014"/>
</CustLoyalty>
</Traveler>
</MatchSubtree>
<NewSubtree>
<Traveler>
<CustLoyalty VendorTypeCode="AL" VendorCode="XX" MembershipID="222258105">
<MembershipLevel MembershipLevelTypeCode="TI" MembershipLevelValue="2"/>
<CustLoyaltyTotals AccountBalance="1100000"/>
</CustLoyalty>
</Traveler>
</NewSubtree>
</ModifySubtree>
</Modify>
</PartialUpdates>
</ProfileInfo>
</Upload></Sabre_OTA_ProfileUploadRQ>

14.4 Extract batch

Customer Insight can generate and send extract files which contain newly created or updated profiles. The frequency of extract files being generated will be agreed upon in advance with the customer so automatic extracts can be scheduled.database

Every file before sending to customer is encrypted via PGP key and transferred via EFG channel.

The extract file contain entire snapshot of the profile stored in CI. To follow PCI compliance rules CI has ability to generate an extract file, which contain PaymentForm (FOP) section in one of the following ways:

- FOP Allow - PaymentForm section is included in the extract file and contain Credit Card and/or Travel Bank details.
- FOP Mask – Payment Form section is sent in a file but Credit Card number/Travel Bank number and ExpireDate is sent as masked.
- FOP Hidden – PaymentForm section is removed from the profiles.

Note The CI implementation team must designate the corresponding settings for the Credit Card display in the extracts according to customer requirements before the activation of the Domain in CI.

Extract sample xml – FOP mask

```
<Sabre_OTA_ProfileExtractRQ Version="6.49" xmlns="http://www.sabre.com/eps/schemas">
  <ResponseMessage>
    <Success/>
  </ResponseMessage>
  <Extract>
    <Profile CreateDateTime="2013-10-24T08:03:44.538Z" PrimaryLanguageIDCode="PL"
UpdateDateTime="2013-10-24T08:03:44.538Z">
      <TPA_Identity ClientCode="AS" ClientContextCode="CI" DomainID="XX"
ProfileDescription="ExtractTestProfile" ProfileName="ExtractTestProfile_TVL"
ProfileNameModifyIndicator="Y" ProfileStatusCode="AC" ProfileTypeCode="TVL"
UniqueID="4154072">
        <Login IsHashed="N" LoginID="TEST_7ND8LP@EXAMPLE.COM"
PasswordHash="677079"/>
        <ProfileSubType SubTypeCode="FFP"/>
      </TPA_Identity>
      <Traveler>
          <Customer>
        <PaymentForm DisplaySequenceNo="1" OrderSequenceNo="1">
          <PaymentCard BankCardVendorCode="AX" CCViewAccess="U"
CardNumber="****" CardTypeCode="CR" EffectiveDate="012010" ExpireDate="****"
ExtendedPaymentNumMonths="5" MaskedCardNumber="7899">
            <CardHolderName>
              <CardHolderFullName>GERMAN BIDDLE</CardHolderFullName>
            </CardHolderName>
          </PaymentCard>
        </PaymentForm>
      </Customer>
    </Traveler>
  </Profile>
 </Extract>
</Sabre_OTA_ProfileExtractRQ>
```

14.5 Customer Insight file name convention.

Files sent to CI should be encrypted with PGP, GPG and sent via secure File Transfer Protocol (SFTP) using Sabre EFG channel.

The following name convention applies:

NameOfSystem_Environment_FileType_AirlineCode/Partner ArlineCode_ProfileTypes_DataOwnerAirlineCode_Date_Time_Number.cmp

Description:

NameOfSystem = CI (for Customer Insight)

Environment = CRT for Certification environment or PRD for Production enviroment

FileType = EX for extract (sent to airline), UP for upload (received from airline)

AirlineCode = 2 character carrierCode /3 character partner airline code i.e. Virgin Australia Partners

ProfileTypes_ = 3 letter profile type (i.e. TVL, AGR, CRP, ALC etc.)

DataOwnerAirlineCode_ = 2 letter airline code for the data owner

YYMMDD_ = Date 2 digit year, 2 digit month, 2 digit day

HHMM_ = Time in 2 digit hour, 2 digit minute (24h based)

Number = three digit number (could be 000)

Sample:

CI_PRD_UP_XX_TVL_FFP_130718_1423_000.pgp – Production file sent to CI

CI_CRT_UP_XX_TVL _FFP_130716_1423_000.pgp – Certification file sent to CI

CI_PRD_EX_XX_TVL_FFP_130718_1423_000.pgp – Production file sent to customer

CI_CRT_EX_XX_TVL _FFP_130716_1423_000.pgp – Certification file sent to customer

Appenidix A 15

15.1 Typical Subject areas and Elements used in Airline standard implementation

PAYLOAD	INTERACT Labels	Notes on data available to move into PNR for reservation
1. <Sabre_OTA_ProfileReadRS xmlns="http://www.sabre.com/eps/schemas"		
2. Version="6.49">		
3. <ResponseMessage>		
4. <Success />		
5. </ResponseMessage>		
6. <Profile CreateDateTime="2010-06-07T14:42:11.675"	Create date/time flag (not visible in GUI)	
7. UpdateDateTime="2010-06-07T14:45:28.782"	Update time (not visible in GUI)	
8. PrimaryLanguageIDCode="EN">	Language (From drop down menu)	
9. <TPA_Identity ClientCode="AS" ClientContextCode="CI"	Indicates AS - Airline Solutions, CI – as Customer Insight system that made the change (ClientContextCode value is not visible in GUI)	
10. UniqueID="4008322259447378" ProfileTypeCode="TVL"	Traveler Profile ID and profile type (TVL)	Moves in to PNR = Profile Index (PI field)
11. ProfileNameModifyIndicator="Y" DomainID="XX"	Partition or Domain for the profile (in this case XX)	
12. ProfileStatusCode="AC">	Profile Status Code- Active.	
13. <Login LoginID="4008322259447378"	Default ID for logging into Sabre Sonic Web (could also use email, etc.)	
14. PasswordHash="8008322411929631" IsHashed="N" />	System generated passcode for logging into Sabre Sonic Web (with option to hash or keep passcode in plain text	
15. <SecurityInfo SecurityQuestionCode="015" SecurityAnswerHash="sophia" />	Security question and answer	
SecurityAnswerHash="sophia" />		
16. <ProfileSubType SubTypeCode="FFP" />	Sub-type = ”WEB” means no FF/registered profile, Subtype = “FFP” indicated enrolled Frequent Flyer	
17. </TPA_Identity>		

18. <Traveler>		
19. <Customer BirthDate="1980-01-01" GenderCode="M">	Birth date and gender code	
20. ServiceSegmentationCode="SVIP">	Service segmentation /labels code (i.e.. VIP or CIP)	
21. <PersonName>		Moves in to PNR = Name filed (-LAST/FIRST, etc.)
22. <NamePrefix>MR.</NamePrefix>	Title	
23. <GivenName>JOSE</GivenName>	First Name	
24. <MiddleName>LUIS</MiddleName>	Middle Name	
25. <SurName>VALENZUELA</SurName>	Last Name	
26. <NameSuffix>JR</NameSuffix>	Suffix	
27. </PersonName>		
28. <Telephone LocationTypeCode="BUS" DisplaySequenceNo="2" OrderSequenceNo="2" InformationText="CL">	Telephone type code = Business from control table dictionary	Moves in to PNR = Phone field (9DFW6802-608-4040 x 21-BUS)
29. <ParsedPhoneNumber CountryCd="56" AreaCd="56"	Country code from drop down of country names (from Parsed phone number country code from control data dictionary) Area code	
30. PhoneNumber="2222222" Extension="11" />	Number and Extension	
31. </Telephone>		
32. <Telephone LocationTypeCode="CEL" DisplaySequenceNo="3" OrderSequenceNo="3" InformationText="US">	Telephone type code = Cell (CEL) from control table dictionary	
33. <ParsedPhoneNumber CountryCd="1" AreaCd="305"	Country code from drop down of country names (from Parsed phone number country code from control data dictionary) Area code	
34. PhoneNumber="2222222" Extension="11" />	Number and Extension	
35. </Telephone>		
36. <Telephone LocationTypeCode="HOM" DisplaySequenceNo="1" OrderSequenceNo="1" InformationText="US">	Telephone type code = Home (HOM) from control table dictionary * there is also "FAX" type in Interact drop down	
37.		
38. <ParsedPhoneNumber CountryCd="1" AreaCd="305"	Country code from drop down of country names (from Parsed phone number country code from control data dictionary) Area code	
39. PhoneNumber="1111111" Extension="21"/>	Number and Extension	

40. </Telephone>		
41. <Email EmailTypeCode="HOM" FormatTypeCode="TEXT"	Email type (Home-HOM) and format type assigned by default by Interact	Moves in to PNR = Email field (PE - email info)
42. EmailAddress="JOSE@EXAMPLE.COM" DisplaySequenceNo="1" OrderSequenceNo="1" />	Email	
43. <Address LocationTypeCode="HOM" DisplaySequenceNo="1" OrderSequenceNo="1">	Address type (available HOM and BUS) from Interact	Moves in to PNR = Address field (W- Name, A-Street Address, C-City, /country - ZIP , etc.)
44. <AddressLine>SUITE 200</AddressLine>	Address line 2	
45. <CityName>MIAMI</CityName>	City	
46. <PostalCd>33133</PostalCd>	Zip / Postal Code	
47. <StateCode>FL</StateCode>	State from Drop down list of states (based on country selection)	
48. <CountryCode>US</CountryCode>	Country code from drop down of country names (from Parsed phone number country code from control data dictionary) Area code	
49. <StreetNmbr>123 MAIN HOME STR</StreetNmbr>	Address line 1	
50. </Address>		
51. <Address LocationTypeCode="CMP" Display SequenceNo="2" OrderSequenceNo="2">	Address type - Business from Interact	
52. <AddressLine>APT 25</AddressLine>	Address line 2	
53. <CityName>LAS CONDES</CityName>	City	
54. <PostalCd>124555</PostalCd>	Zip / Postal Code	
55. <CountryCode>CL</CountryCode>	State from Drop down list of states (based on country selection)	
56. <StreetNmbr>ALONSO DE CORDOBA 210</StreetNmbr>	Address line 1	
57. </Address>		
<PaymentForm DisplaySequenceNo="1" OrderSequenceNo="1">	Payment information including Travel Bank for service credits	Can be used to move in form of payment
<PaymentCard CardTypeCode="TB" BankCardVendorCode="BT"		
CardNumber="8982222228232" MaskedCardNumber="8232"		
CCViewAccess="Y" ExtendedPaymentNumMonths="0">		

<CardHolderName>		
<CardHolderFullName>Travel Bank		
Name</CardHolderFullName>		
</CardHolderName>		
<CardIssuerName IssuerName="Sabre Travel Bank" />		
</PaymentCard>		
</PaymentForm>		
58. <Document DocID="854236" DocTypeCode="PSPT"	Document information (i.e.. Passport) can include issued / expiration dates, country, names, etc.	Can be set to move as SSR for FOID (form of Identification)
59. DocIssueCountryCode="CL" OrderSequenceNo="1" />		
60. <Document DocID="69741258" DocTypeCode="DRLS"	Document information (i.e.. Driver license) can include issued / expiration dates, country, names, etc.	
61. DocIssueCountryCode="CL" OrderSequenceNo="2" />		
62. <CustLoyalty VendorTypeCode="AL" VendorCode="XX"	Frequent Flier number for the hosted carrier. (Can include mileage balance and various other fields such as membership date, expiration date, etc.	Moves in to PNR = Frequent Flier (FF) including ability to display tiers for hosted loyalty program and airline alliance (can include registration date, mileage balance, awards, etc.
63. MembershipID="25038665" OrderSequenceNo="1">		
64. <MembershipLevel MembershipLevelTypeCode= "TI" MembershipLevelValue="02"		
65. AllianceCode="*O" AllianceLevelValue="02" />		
66. <CustLoyalty VendorTypeCode="AL" VendorCode="BA"	OA numbers, Airline	
67. MembershipID="33456565" OrderSequenceNo="2"/>	Number	
68. <CustLoyalty VendorTypeCode="AL" VendorCode="AA "	OA numbers, Airline	
69. MembershipID="12344546" OrderSequenceNo="3"/>	Number	
71. <CustLoyalty VendorTypeCode="AL" VendorCode="IB"	OA numbers, Airline	

72. MembershipID="34354534" />	Number	
73. <CustLoyalty VendorTypeCode="AL" VendorCode="MX" MembershipID="32344 56" />	OA numbers, Airline	
74.	Number (if customer is Frequent flyer would also have Membership Level and CustomerLoyaltyTotals)	
75. </Customer>		
76. <PrefCollections>		
77. <AirlinePref TripTypeCode="CP">	Set behind scenes by Interact	
78. <AirlineSeatPref PreferLevelCode="P">	Indicates "Preference: Set behind scenes by Interact	
79. <SeatInfo SeatPreferenceCode="NSSA" />	Seat Preference	Can be set to select seat assignments
80. </AirlineSeatPref>		
81. </AirlinePref>		
82. <AirlinePref TripTypeCode="AZ" />	All (preference)	
83. <TPA_MarketingPreference>		
84. <Consent TypeCode="OI2" Value="Y" ApplicationCode="AS" />	Check boxes to designate certain marketing preferences (e.g. subscription opt-in or offer preferences)	
85. </TPA_MarketingPreference>		
86. </PrefCollections>		
87. <TPA_Extensions>		
88. <Remark Text="VIP TRAVELER REMARK" TypeCode="DP"	Remarks (free text)	Can be set to move to PNR as SSR (Special Service Request) or Remark line
89. DisplaySequenceNo="1" OrderSequenceNo="1" />		
90. <Remark Text="DENIED BOARDING ON JUNE 20 10" TypeCode="DP"	Remarks (free text)	
91. DisplaySequenceNo="2" OrderSequenceNo="2 " />		
92. <SSR SSRCode="BLND" TypeCode="AFAX" Text=" "	SSR - Request	Can be set to move to PNR as SSR (Special Service Request) or Remark line
93. DisplaySequenceNo="1" OrderSequenceNo="1" />		

94. <SSR SSRCode="XBAG" TypeCode="AFAX" Text=" "	SSR - Request	
95. DisplaySequenceNo="2" OrderSequenceNo="2" />		
96. <BusinessSystemIdentityInfo SystemName="L OYALTYLID" ID="81965971" OrderSequenceNo="1 " />		
97. <BusinessSystemIdentityInfo SystemName="RNTID" ID="1232474" OrderSequenceNo="2" />		
98. <BusinessSystemIdentityInfo SystemName="EID" ID="1551740" OrderSequenceNo="3" />		
99. <CustomDefinedData CustomFieldCode="AFE" Valu e="Service incidents ranking 1" DomainID="XX" DisplaySequenceNo="1" OrderSe quenceNo="1" />	Can define custom fields for individual airline use if need fields not defined in Profiles systems data model	
100. <EnrollmentInfo EnrollmentChannel="WEB" EnrolledStatus="false" />		
101. </TPA_Extensions>		
102. </Traveler>		
103. </Profile>		
104. </Sabre_OTA_ProfileReadRS>		
** This is intended only for illustration purposes.		