

**ConversationId Technical Documentation**

**October 7th, 2016**

**Revision tracking**

|  |  |  |  |
| --- | --- | --- | --- |
| **VERSION #** | **DATE** | **ORIGINATOR** | **CHANGE DESCRIPTION** |
| 4.0 | November 25,2016 |  | Updates after review. |
| 3.0 | November 23,2016 |  | Added version support and clarification about generation of attributes. |
| 2.0 | November 7,2016 |  | Updated approach with TrxId. |
| 1.0 | October 7, 2016 |  | Initial version |

1. ConversationId 4

2. Typical ConversationId usage scenarios 5

3. Format of ConversationId field 5

4. Additional considerations 6

4.1 Multiple shopping requests for single customer shop 6

4.2 Caching on agency side 6

4.3 Working with Meta search sites 7

4.4 Long running customer sessions 7

4.5 Blind sells 7

5. Benefits 7

# ConversationId

ConversationId is a field in SOAP Header that is meant to tie together messages within the context of the same “conversation” – meaning logical user session. Here is an example of the message header with ConversationId field.

 *<SOAP-ENV:Header>*

 *<eb:MessageHeader SOAP-ENV:mustUnderstand="1 eb:version="2.0">*

***<eb:ConversationId >280b16ec-5eac-46c0-893f-c88f8e8cb632</eb:ConversationId>***

 *<eb:From>*

 *<eb:PartyId type="urn:x12.org:IO5:01">ABC</eb:PartyId>*

 *</eb:From>*

 *<eb:To>*

 *<eb:PartyId type="urn:x12.org:IO5:01">ABC</eb:PartyId>*

 *</eb:To>*

 *<eb:CPAId/>*

 *<eb:Service eb:type="OTA">BargainFinderMaxService</eb:Service>*

 *<eb:Action>BargainFinderMax\_ADRQ</eb:Action>*

 *<eb:MessageData>*

 *<eb:MessageId>1919</eb:MessageId>*

 *<eb:Timestamp>2016-08-12T08:34:43</eb:Timestamp>*

 *<eb:TimeToLive>2016-08-13T08:34:43</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 valueType="String" EncodingType="wsse:Base64Binary">Shared/IDL:IceSess\/SessMgr:1\.0.IDL/….</wsse:BinarySecurityToken>*

 *</wsse:Security>*

 *</SOAP-ENV:Header>*

Until now Sabre was not enforcing nor using ConversationId value in any special way leaving decisions how to use that field to API users.

As an attempt to standardize and improve internal analytical capabilities, Sabre is proposing particular design pattern for ConversationId usage. ConversationId is part of a SOAP header and should be sent together with all Sabre Web Services requests within customer session (Shopping, Booking, Pricing, Reservation Management, etc.). The remaining part of the document contains usage suggestions that need to be implemented on the API consumer part.

Typical ConversationId usage scenarios

In typical interaction end customer opens an agency Web page and is assigned a Web application session. Within that Web application session, he/she makes multiple Shopping request followed by Booking request of one of the returned itineraries. Within the same session, there might be subsequent Sabre Web Services requests (e.g. AddRemarks or AddPsgDetails).



Format of ConversationId field

ConversationId attribute is used to pass SessionId, TrxId and RefToTrxId values.

SessionId is unique for logical end user session. TrxId is unique for each transaction. RefToTrxId is not used on Shopping transactions; it is used on Booking transaction to designate the Shopping transaction that returned the itinerary we are trying to book. It is important to remember that SessionId is not associated in anyway with technical Sabre Web Services session.

 The format of the ConversationId is Version+”@”+SessionId+”@”+TrxId+”@”+ RefToTrxId. All part of ConversationId are optional. Version is to be set to “V1” currently. SessionId, TrxId are generated by agency, and TrxId should be unique across all agency requests (not only within the Session).

Examples:

V1@280b16ec-5eac-46c0-893f-c88f8e8cb632***@***310b16ec-5dad-46c0-893f-c88f8e8cb643***@***

V1@280b16ec-5eac-46c0-893f-c88f8e8cb632***@***310b16ec-5dad-46c0-893f-c88f8e8cb643***@***780b16ec-5eac-46c0-893f-c88f8e8cb699

Additional considerations

Multiple shopping requests for single customer shop

When OTA sends multiple BFM requests for triggered by single end traveler request, in case those request return overlapping itineraries subsequent booking request might use any of TrxId for booking of the itinerary that is shared between responses.

Scenario:

Traveler A shops on OTA. OTA sends two BFM requests to Sabre:

ShoppingReq1(SessionId=1, TrxId=1)

Returns: Itin1, Itin2, Itin3

ShoppingReq2(SessionId =1, TrxId=2)

Returns: Itin2, Itin4

When we want to book

1. Itin1 - we need to send BookingReq(SessionId =1,TrxId=3,RefToTrxId=1)
2. Itin4 - we need to send BookingReq(SessionId =1,TrxId=3,RefToTrxId=2)
3. Itin2 - we can send BookingReq(SessionId =1,TrxId=3,RefToTrxId=2) or BookingReq(SessionId =1,TrxId=3,RefToTrxId=1)

Caching on agency side

Within OTA cache there is a need to store TrxId of the shopping request that populated the cache and use it on subsequent Booking transaction.



Scenario:

Traveler A charges the cache with his Shopping transaction. The cached results are later used by Traveler B to perform the booking.

Traveler A shops:

ShoppingReq1(SessionId =A, TrxId=1)

Returns: Itin1, Itin2, Itin3

*These three itineraries are stored into the cache.*

Traveler A stops shopping on OTA.

Traveler B starts its session:

ShoppingReq2(SessionId =B, TrxId=2)

Returns: Itin4, Itin5, Itin6

Then he executes the shop that returns Itin1, Itin2, Itin3 from OTA cache.

When Traveler B wants to make a booking, we need to send:

1. Itin1 – BookingReq1(SessionId =B, TrxId=3,RefToTrxId=1)
2. Itin4 – BookingReq2(SessionId =B, TrxId=4,RefToTrxId=2)

Usage of Sabre Shopping Cache is not impacting the way ConversationId is to be used – the customer using Sabre Cache should treat such requests as a standard BFM request in this context.

Working with Meta search sites

If OTA is able to associate redirected customers with initial meta search it will be beneficial to use the same SessionId on both Meta initiated request as well as the request that are sent after redirect. In case, such association is not possible keeping separate SessionIds or not using SessionId on Meta requests is acceptable. The diagram below assumes meta requests cannot be identified as belonging to the same traveler session. The alternative situation will allow OTA to combine XYZ and ABC sessions into one.



Long running customer sessions

For OTA WebApp session spanning longer period (e.g. days) there is no need to maintain the same SessionId on Shopping transactions across multiple days.

Blind sells

For blind sell, we might or might not have SessionId but we will not have RefToTrxId so Booking transaction will be executed with empty RefToTrxId attribute.

Benefits

Enhancing OTA’s end-to-end flow by adding ConversationId will benefit both customers and Sabre.

Key benefits for customers:

1. Improvement in conversion management by
	1. Intelligent airline polling control
	2. Support in customer A/B testing capabilities
	3. Analyzing conversion parameters per product/service
	4. Shopping Cache fine tuning
	5. Accurate failure rate management (identifying blind sell/cache issues)