Oracle® Hospitality OPERA Property Management

Developer's Guide
5.4.x.x and 5.5.x.x
E89601-01

July 2017

ORACLE

Contents

Preface

This document provides all of the information necessary to develop custom applications that interact with OWS Web Services through the programming language of your choice. This functionality is provided through standard Web Services technologies. As of OPERA Version 5.0, we have added new functions to allow for developers to create a Kiosk from the OWS web Services. This combined platform should provide our partners with an even broader base of tools to allow connectivity to the changing face of today's Guest.

Audience

This document is intended for web developers, programmers, and architects.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received and any associated log files
- Screen shots of each step you take

Documentation

Oracle Hospitality product documentation is available on the Oracle Help Center at http://docs.oracle.com/en/industries/hospitality/

Revision History

Date	Description of Change
July 2017	• Initial publication

What are Web Services?

Web Services are applications whose logic and functions are accessible using standard Internet protocols and data formats such as Simple Object Access Protocol (SOAP) over Hypertext Transfer Protocol (HTTP). A Web Services Description Language (WSDL) file defines the application programmatic Interface (API) and tools are used to generate API stubs in the language of choice. It is the Web Services (or SOAP) tools that are used to generate the stubs that then send SOAP formatted method calls to the server to enable widespread inter-operability.

Similar to component-based development, Web Services represent black-box functionality that can be reused without worrying about how the service is implemented. A Web Service interface is defined strictly in terms of the messages that the service accepts and generates. Applications using a Web Service can be implemented on any platform in any programming language as long as they can create and consume messages defined for the service interface. A Web Service can also aggregate other services to provide a higher-level set of features.

Why a Developer should be interested in Web Services

- **Inter-operability** - Any Web Service can interact with any other Web Service and can be written in any language.
- **Communication** - Web Services communicate using HTTP and XML. Any connected device that supports these technologies can both host and access Web Services.
- **Easy to Understand** - The concepts behind Web Services are easy to understand and developers can quickly create and deploy them using many tools available on the Web.
- **Industry Support** - Major content providers and vendors support the Web Services movement.

The Simple Object Access Protocol (SOAP), a method of sharing messages between client and server, was developed to deal with the limitations of XML/HTTP. SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework, providing a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics.

1 Tools and Platforms

Oracle provides a Web Services Description Language (WSDL) file from which you will need to generate stubs in the programming language of your choice. A SOAP toolkit for your particular programming language will generate the required stubs and SOAP message for each method call to the server. Be forewarned that there are new tools being developed all the time, and innovation is very rapid in this area. Most standard programming languages are represented.

SOAP allows third-party developers to interface with the OPERA Web Suite (OWS) product using XML data files over the Hypertext Transfer Protocol (HTTP) protocol. SOAP structures these XML files into remote procedure calls.

Please see our .wsdl files for more information on the contents of OWS SOAP requests and responses. The reference section provides the semantics for calling each routine.

Java Environment

This build.xml file assumes you have installed the Axis toolkit. This code also assumes

that you have setup the correct class path. The code can be compiled using J2SE 1.4.2.

The code generated for Java web services client is generated by using WSDL2Java:

java org.apache.axis.wsdl.WSDL2Java -o .\ -T 1.1
Nhttp://tempuri.org/=com.micros.webservices -S true
https://yourdomain.com/yourows/yourwebservice.asmx?wsdl

This will create namespace folders with the locator/stub code inside. You need to replace the http://tempuri.org with the corresponding Action URI located in AvailabilityServiceLocator.java.

```
// Use to get a proxy class for AvailabilityPort

private java.lang.String AvailabilityPort_address =
"https://yourdomain.com/yourows/yourwebservice.asmx";
```

For additional information, see update.bat, the generated code in the \Proxies\Java folder, and code samples in the \Samples\Java folder.

The SOAP header is not automatically generated into the locator class as in .NET. It exists in \og_4_3\Core\OGHeader.java. To initialize, setup the OGHeader and pass it to the function

`void` **setHeader**(SOAPHeaderElement header) from org.apache.axis.client.Stub before any calls to the Web Service's method.

C# Environment

This code was created with Visual Studio .Net. Web Services Enhancements 2.0 (WSE) was utilized for the WS-Security sample.

1. Install WSE on your development machine.
2. Create a new C# Windows Application solution in Visual Studio 2003.
3. Under menu Project-Add Web Reference, generate a proxy from the selected service.WSDL.
4. Edit the generated Reference.cs file and change the tempuri.org URL with the SOAP Action URI of the .asmx file:public AvailabilityService()

   {this.Url ="**https://yourdomain.com/yourows/yourwebservice.asmx**";}.
5. Copy the sample code or create your own code using the new proxy libraries.

You can also use the wsdl.exe tool from .NET Framework SDK to generate the client code. For example, with an availability service:

wsdl /protocol:SOAP /namespace:OWS.Availability
http://webservices.micros.com/ows/5.1/Availability.wsdl

This generates in the same folder as the AvailabilityService.cs class that you can import into your project. Follow step 4 (above) to change the default URI.

For additional information, see the code generated in \Proxies\C#.

Visual Basic .NET Environment

This code was created using Visual Studio 2003 with Web Services Enhancements (WSE) 2.0.

The following steps are used to create the sample application:

1. Install WSE on your development machine.
2. Create a new Visual Basic Windows Application solution in Visual Studio 2003.
3. Under the menu Project-Add Reference->Add reference to Microsoft.Web.Services.dll.
4. Under the menu Project-Add Web Reference, generate a proxy from the selected service .WSDL.
5. Edit the generated Reference.vb file and change the tempuri.org URL with the SOAP Action URI of the .asmx file.

   Public Sub New()

   MyBase.New

   Me.Url = " **https://yourdomain.com/yourows/yourwebservice.asmx**"

   End Sub
6. Because of the Microsoft Visual Basic Deficiency, follow the Microsoft procedures in Knowledge Base 815211. The proxy file (Reference.vb) must also be edited for out parameters, as described the in KB815211 article.
7. Copy the sample code, or create your own code using the new proxy libraries.

You can also use wsdl.exe tool from .NET Framework SDK to generate the client code. For example, with availability service:

wsdl /protocol:SOAP /language:VB /namespace:OWS.Availability http://webservices.micros.com/ows/5.1/Availability.wsdl

This will generate in the same folder AvailabilityService.vb class that you can import into your project and follow from step 4 (above) to change the default URI. For additional information, see the generated code in \Proxies\VB.NET folder, and code samples under \Samples\VB.NET.

2 OPERA Web Suite (OWS)

A Brief Description of the Functions

OWS currently includes the following services and functions:

Activity Service Functions

The following functions are currently provided through the Activity Web Service:

Function	Description
ActivityLookup	Retrieves a list of activities the guest has reserved.
ActivityAvailability	Retrieves available activities.
CancelActivity	Cancels guest activities.
CreateActivity	Creates guest activities.
FetchActivity	Fetches guest activities.

Availability Service Functions

The following functions are currently provided through the Availability Web Service:

Function	Description
Availability	Retrieves lists of available rooms and rates. The user should provide at least a date range, hotel criteria, number of rooms, total guests, and number of children. Rate range or plans, rate tiers, alternate date searches, and search by membership number, specific room type or block criteria are also allowed. Setting the summaryOnly flag to true results in a General Availability request and setting the flag to false results in a Detail Availability request.

FetchAvailableItems	Retrieves a list of available add-on items by date. The user should provide at least the date range (a maximum of seven days), hotel criteria, and the item class or group. Specifying an item group retrieves a list of available items in that group. Setting the isPackage flag to true adds attached packages to each item in the list and setting the flag false ignores package information for items.
FetchAvailablePackages	Retrieves a list of available add-on packages and package groups by date. The user should provide at least a date range and hotel criteria. Confirmation identifier, leg confirmation identifier, number of rooms, and counts of adults and children are also allowed.
FetchCalendar	Retrieves a list with room occupancy, rate, and restriction details by date. The user should provide at least a date range and hotel criteria. Room type or block, adult or child guest counts, and rate plan information are also allowed.
FetchExpectedCharges	Retrieve a rate, taxes and package breakdown for stay for requested stay date, rate code, room type, property.
FetchItemGroups	Retrieves a list of available add-on items. Hotel criteria must be provided. Setting flag groupOnly to true (the default) retrieves the list of item groups and setting the flag to true retrieves a list of all available items.
GDSAreaAvailability	Fetches the availability status of a list of properties submitted in the request message. The response contains the property status, chain code, Property ID and the minimum and maximum rate ranges dynamically retrieved from the OPERA database.
RegionalAvailability	Retrieves a list of hotels. The user must provide at least a geographic region code or hotel criteria, date range, and the number of guests. Number of rooms and a rate range are also allowed.
RegionalAvailabilityExt	Retrieves a list of and provides extended information about hotels. The user must provide at least a geographic region code or hotel criteria, date range, and number of guests. Number of rooms and a rate range are also allowed.

Brochure Service Functions

The following functions are currently provided through the Brochure Web Service:

Function	Description
SendBrochure	Sends a brochure, given the hotel criteria, brochure type, name identifier, and e-mail address.

Guest Services Functions

The following functions are currently provided through the Guest Services Web Service:

Function	Description
FetchOptInSetup	Fetch Optin setup fields for requested resort.
UpdateReservationForOptIn	Update Reservation with OptIn Data.
UpdateRoomStatus	Update room status for requested reservation.
WakeUpCall	Retrieves the wakeup calls for a reservation. In addition, it can add wakeup calls and delete wakeup calls for the reservation. This function can only be utilized when a reservation is "IN HOUSE."

Housekeeping Functions

The following functions are currently provided through the Housekeeping Web Service: Allows for the inputting of housekeeping information via web services.

Function	Description
FetchHouseKeepingDiscrepancies	Displays all rooms where the number status conflicts between Front Office and Housekeeping.
FetchHouseKeepingStatistics	Displays the housekeeping statistics information.
FetchHouseStatus	The House Status is a survey of all movements for the current date as well as any future date. Information available from this includes arrivals, departures, available rooms, housekeeping status, and expected occupancy for the date searched.

UpdateHouseKeepingDiscrepancies	Updates the HouseKeeping Discrepancies with actual status.

Information Service Functions

The following functions are currently provided through the Information Web Service:

Function	Description
CurrencyConverter	Returns the converted amount, given the original currency code, original amount, converted currency code, and hotel criteria. An exchange type is also allowed.
GetScreenItems	Retrieves all the screen items.
QueryAwardsSchedules	Returns Awards Schedules for Rate, Product and Upgrade awards types.
QueryChainInformation	Returns information about the chain, given the chain code. Information returned includes addresses, emails, phone numbers, marketing text, loyalty program details, booking conditions, and frequent flyer program details. This functionality is only available when in MyFidelio mode.
QueryHotelInformation	Retrieves information about a hotel, given the hotel criteria. Contact and location details as well as information about facilities, amenities, services, and alternative properties are provided.
QueryLov	Retrieves a list of values, given the query type string. A wide variety of resort configuration details are viewable.
QueryPackageItems	Returns package groups/packages/item groups/items setup in the resort.
QueryProductItems	Returns product groups/products/item groups/items setup in the resort.
QueryRate	Retrieves rate information, given the hotel and rate codes. A date range is also allowed. The information includes policies, requirements, restrictions, and other descriptive details.

Meeting Room Service Functions

The following functions are currently provided through the Meeting Room Web Service:

Function	Description
CreateBlock	Creates a block in OPERA.
MeetingAvailability	Single-property general or detailed availability for function space.
MeetingCancelEvent	(Not Implemented yet) Deletes existing event.
MeetingCreateEvent	Create a business block and event which includes meeting room, miscellaneous Items, Menu, additional menu items and sleeping rooms.
MeetingCreatePackageEvent	Create a business block with a catering package which includes Package ID, additional miscellaneous Items, menu items, and sleeping rooms.
MeetingFetchBlockDelegates	Retrieves the delegates (attendees) for a business block or event.
MeetingFetchContract	Fetches the CONTRACT, BEO or INQUIRY report and returns it as a link or emails it as a PDF.
MeetingFetchEvent	Fetch created event with the ability to return sleeping rooms, meeting room, miscellaneous Items, menu, and menu items as well as a breakdown of the event charges.
MeetingFetchMenu	Returns configured menus and related menu items depending on the request. The data returned will be for one property.
MeetingFetchMenuItem	Depending on the request, returns configured menu items. The data returned will be for one property.
MeetingFetchMiscellaneousItem	Returns all configured miscellaneous items for one property.

MeetingFetchMyRegisteredEvents	Retrieves an attendee's registered events.
MeetingFetchPackageEvent	Fetch created packaged event will have the ability to return sleeping rooms, catering package details, meeting room, miscellaneous Items, menus, additional menu items and miscellaneous Items as well as a breakdown of the event charges.
MeetingModifyEvent	Allows for modifying existing sleeping room and non-catering package event details as well as adding or deleting events and resources.
MeetingModifyPackageEvent	Allows for modifying existing sleeping room and catering package details for an existing business block. Also allows for adding or deleting non-package resources.
MeetingMultiPropertyAvailability	Multi-property availability will return all properties and function space that meet the requirements, regardless of Function Space availability.
MeetingPackageAvailability	Returns general or detailed package availability for a property.
MeetingRegisterEventAttendees	Registers attendees for an event.
MeetingRoomCopyBlock	Copies an existing business block to a new date along with new business block data.
MeetingRoomCreateRelationship	Creates relationship between a contact profile and other profile types.
MeetingRoomFetchBlock	Returns the business block details including events, package events and sleeping rooms.
MeetingRoomFetchMyBlocks	Returns a list of business blocks by either ContactNameId or AccountNameId.

ModifyBlock	Modify a business block by adding notes, changing name, adding/deleting number of rooms, changing number of days, and changing room types.

Membership Service Functions

The following functions are currently provided through the Membership Web Service:

Function	Description
AddBenefit	Applies benefits to the member who will be automatically upgraded to the level associated with the benefit code, given the membership ID.
AddPromoSubscription	Adds a promotional subscription to a membership, given the membership identifier and promotion information.
CancelConsumedPoints	Cancels points redeemed/consumed when a reservation is made and then cancelled. For example, if a member redeems points while making a reservation, and then cancels the reservation, the points redeemed will be returned.
CancelECertificate	Cancel an e-certificate.
ConsumeECertificate	Consume an issued e-certificate.
ConsumePoints	Adds customer points and returns the transaction identifier, given the membership identifier and loyalty award program request details.
ConsumePointsOthers	Adds customer points and returns the transaction identifier, given the membership number and product award details.
DeletePromoSubscription	Deletes a promotional subscription from a membership, given the membership identifier and promotion information.
FavoriteGuest	Add/Delete/Fetch favorite guest to/from a profile.
FetchAvailableECertificates	Returns all configured E-Certificates for a date range.

FetchBenefits	Returns all benefit codes that are applied to a member.
FetchECertificates	Fetch all e-certificates issued to a profile.
FetchEnrollmentCode	Retrieves the enrollment code for a membership, given the membership identifier.
FetchMemberPoints	Returns member points information.
FetchMemberTierWizard	Returns information on the members upgrade requirements or the membership level on downgrade which will be evaluated based on the inputs of a future evaluation date and membership ID.
FetchMembershipTransactions	Retrieves a list of membership transactions, given the membership identifier or membership record.
FetchNextCardNumber	Retrieves a membership card number, given the membership type.
FetchPointsExchange	Retrieves the membership type points based on the exchange rate setup between the two membership types.
FetchProductAwards	Retrieves a list of product awards in effect, given the membership type and level, date range, and number of rooms. Details on the product award requirements are provided.
FetchPromoSubscriptions	Retrieves a list of promotional subscriptions attached to a membership, given the membership identifier.
FetchRateAwards	Retrieves a list of rate awards in effect, given the membership type and level, date range, number of rooms, points range minimum and maximum values, and a list of hotel references. Details on the rate award requirements are provided.
FetchStatement	Retrieves a hotel loyalty program statement for a membership, given the membership identifier and the statement identifier and date. Summary information and details on the membership transactions are provided.
FetchStatementRefs	Retrieves a list of statement records, given the membership identifier or membership record. Each record includes the statement identifier and date.

FetchTransactionAwards	Fetch Transaction Award Codes.
FetchUpgradeAwards	Retrieves a list of award upgrades in effect, given the membership type and level, date range, and number of rooms. Details on the upgrade and its requirements are provided.
IssueECertificate	Issue e-certificate to a profile.
IssueTransactionAward	Issue or Redeem Transaction Awards points.
ReIssueMemberCard	Member will be scheduled for next fulfillment export for code REISSUE NEW CARD.
TransferPoints	Transfer points from one membership type to another membership type.
UpdateEnrollmentCode	Adds or modifies the enrollment code of a membership, given the membership identifier and new enrollment code.

Name Service Functions

The following functions are currently provided through the Name Web Service:

Function	Description
DeleteAddress	Deletes a specific address from a profile, given the address identifier.
DeleteComment	Deletes a specific comment from a profile, given the name and comment identifiers.
DeleteCreditCard	Deletes a specific credit card record from a profile, given the credit card identifier.
DeleteEmail	Deletes a specific e-mail record from a profile, given the e-mail identifier.
DeleteGuestCard	Deletes a specific membership number from a profile, given the guest card identifier.
DeletePassport	Deletes the passport information from a profile, given the name identifier.

DeletePhone	Deletes a specific phone record from a profile, given the phone record identifier.
DeletePreference	Deletes a specific preference from a profile, given the name identifier and the preference.
DeletePrivacyOption	Deletes the privacy option settings for a profile, given the name identifier.
FetchAddressList	Retrieves a list of address records from a profile, given the name identifier.
FetchClaimsStatus	Fetches the status of the claim submitted by a profile.
FetchCommentList	Retrieves the list of comments from a profile, given the name identifier.
FetchCreditCardList	Retrieves the list of credit card records from a profile, given the name identifier.
FetchEmailList	Retrieves the list of e-mail address records from a profile, given the name identifier.
FetchGuestCardList	Retrieves the list of membership records from a profile, given the name identifier.
FetchName	Retrieves the name record from a profile, including any birthday and gender information, given the name identifier.
FetchNameUDFs	Retrieves the list of user-defined field values from a profile, given the name identifier.
FetchPhoneList	Retrieves the list of phone records from a profile, given the name identifier
FetchPreferenceList	Retrieves the list of preferences from a profile, given the name identifier.
FetchPrivacyOption	Retrieves the list of privacy option settings for a profile, given the name identifier.

FetchProfile	Sends profile request to OPERA and OPERA returns all of the profile information in the response message.
FetchProfileBenefits	Fetch all promotions and e-certificates associated with a profile.
FetchSubscription	Retrieves all the external system NameIds of the profile.
GetPassport	Retrieves the passport information from a profile, given the name identifier.
InsertAddress	Adds an address record to a profile, given the name identifier and address information to insert.
InsertClaim	Submits a claim for a profile.
InsertComment	Adds a comment to a profile, given the name identifier and comment information to insert.
InsertCreditCard	Adds a credit card record to a profile, given the name identifier and credit card information to insert.
InsertEmail	Adds an e-mail address record to a profile, given the name identifier and e-mail address to insert.
InsertGuestCard	Adds a membership record to a profile, given the name identifier and membership information to insert.
InsertPhone	Adds a phone record to a profile, given the name identifier and phone information to insert.
InsertPreference	Adds a preference to a profile, given the name identifier and preference description.
InsertUpdateNameUDFs	Adds to or modifies the list of user-defined field values in a profile, given the name identifier and a list of UDF records.
InsertUpdatePrivacyOption	Adds to or modifies the list of privacy option settings for a profile, given the name identifier and a list of privacy option setting records.

NameLookup	Retrieves a list of profile records, given the name look-up credit card or membership criteria.
RegisterName	Creates a new profile and its name identifier, given at least the person's name record. Native name record, birth date, gender, address record, phone record, passport record, login name, and login password are also allowed. Specifying the login information also creates a web user.
TravelAgentLookup	Allows the look up of travel agent given the NameID (IATA Number). Returns the travel agent's company name, address, phone number, email address, and IATA number in the response.
UpdateAddress	Modifies an address record in a profile, given the address information.
UpdateClaim	Adds comments to an existing claim.
UpdateComment	Modifies a comment record in a profile, given the name identifier and comment information.
UpdateCreditCard	Modifies a credit card record in a profile, given the credit card information.
UpdateEmail	Modifies an e-mail address record in a profile, given the-e-mail address information.
UpdateGuestCard	Modifies a membership record in a profile, given the membership information.
UpdateName	Modifies the name information in a profile, given at least the name identifier and person's name record. Native name record, birth date, and gender are also allowed.
UpdatePassport	Adds or modifies the passport record in a profile, given the name identifier and passport information.
UpdatePhone	Modifies a phone record in a profile, given the phone information.

Reservation Service Functions

The following functions are currently provided through the Reservation Web Service:

Function	Description
AccompanyGuest	Attaches an accompanying guests to a reservation. Requires 2 calls: The first call creates the booking and receive the response for the primary guest and the second call attaches the accompanying guests. This function requires the Reservation group application function ACCOMPANYING GUEST to be enabled in OPERA PMS.
AssignRoom	Assigns the room number for a reservation, given at least the hotel criteria and reservation identifier. Requested room number and station identifier are also allowed.
BookHoldItems	Attaches held inventory items to a reservation, given the hotel criteria, confirmation or reservation identifier, and the list of hold item identifiers to attach.
BreakShare	Break a shared reservation.
CancelBooking	Cancels a reservation and returns a cancellation identifier, given at least the hotel criteria, confirmation identifier, and cancellation type summary. GDS identifier and Leg confirmation identifier are also allowed.
ClearItemHold	Returns a held item to available inventory, given the hotel criteria and hold item identifier.
CombineShare	Share two reservations.
ConfirmBooking	Confirms a booking for a GDS channel type, given hotel reference information and the booking confirmation number, and the GDS ID (optional). If this is a multi-leg booking, the leg number is also allowed. The response includes the result and confirmation number. Used only when Session Control is active in Channel setup.

CreateBooking	Creates a new reservation and returns confirmation and reservation identifiers, given at least hotel reservation criteria including room stay, guest information, and confirmation instructions. Reservation history information, user defined values, and bill header are also allowed. Multiple rate codes, including multiple child rates, are supported. CreateBooking also allows packages to be attached to the reservation during the booking process.
CreateItemHold	Creates a hold item and returns a hold item identifier, given the hotel criteria, item code, quantity, and date range.
CreateMultipleBookings	Create multiple bookings.
DeleteAccompanyGuest	Removes an accompanying guests from a reservation. This function requires the Reservation group application function ACCOMPANYING GUEST to be enabled in PMS.
DeleteInventoryItem	Removes inventory attached to a reservation, given at least the hotel criteria, confirmation or reservation identifier, and item or item group representing a collection of items to remove. Leg confirmation identifier is also allowed. If a single deleted item is the sole element of a package, the package is also removed from the reservation.
DeletePackages	Removes an add-on package from a reservation, given at least the hotel criteria, confirmation identifier, and product code. Leg confirmation identifier is also allowed.
EmailConfirmation	Sends a reservation confirmation, given the confirmation identifier and e-mail address.
FetchAvailableOffers	Fetch available Upsell offers for reservation.
FetchBookedInventoryItems	Retrieves the list of inventory items attached to a reservation, given at least the hotel criteria and confirmation or reservation identifier. Leg confirmation identifier is also allowed. Details including the item code and quantity are provided for each date.

FetchBookedPackages	Retrieves the list of packages and package groups attached to a reservation, given at least the hotel criteria, and confirmation identifier. Leg confirmation identifier is also allowed. Details about each package or package group and expected charges by date are provided.
FetchBooking	Retrieves a reservation, given at least the confirmation identifier or GDS identifier or customer reference identifier. Hotel criteria, leg confirmation identifier, and external hotel reference and leg confirmation number are also allowed. Details are provided on the room stay, guest profile, and confirmation instructions.
FetchBookingForPointUpdate	Retrieves a custom list of reservations for assigning consumed membership points, given the hotel criteria.
FetchHoldItems	Retrieves a list of items on hold, given the hotel criteria and a comma-separated list of hold item identifiers. The item code, quantity, and date range of the held items are provided.
FetchRoomUpgrades	Fetch Room Upgrade available for member reservation.
FetchSummary	Retrieves a reservation, given the confirmation identifier. Summary information is provided on the status, room stay, and guest profile.
FutureBookingSummary	Retrieves a list of reservations for future dates, given some filtering criteria. The filtering may be by date range (booked or arrival date), name or corporate identifier, last name, first name, credit card number, search level (subgroup, group, or booker), and/or other defined filters (creation date, contract identifier, membership record, hotel criteria, confirmation identifier, reservation identifier, and key track data). Source and Origin codes for reservations can also be returned in the response.
GetReservationStatus	A lightweight operation that returns selected items to verify a reservation's current status.
GuestRequests	Add or remove or update or fetch comments, traces, specials attached to a reservation.

IgnoreBooking	Ignores (cancels) a booking for a GDS channel type, given hotel reference information and the booking confirmation number, and the GDS ID (optional). If this is a multileg booking, the leg number is also allowed. The response includes the result and confirmation number. Used only when Session Control is active in Channel setup.
ModifyBooking	Modifies a reservation, given the updated reservation record. The confirmation identifier in the record is used to fetch the existing reservation. Key track data is also allowed. Multiple rate codes, including multiple child rates, are supported. ModifyBooking also allows packages to be attached or updated during modification of the booking. In addition, moving a reservation from one resort to another with the same change is also supported.
ModifyItemHold	Modifies a held item, given the hotel criteria, hold item identifier, and the new quantity and/or date range values.
PreCheckin	A lightweight operation that allows modification of the selected items needed for pre-check-in.
ReInstateReservation	Reinstate a cancel/noshow reservation to reserved.
ReleaseRoom	Releases the assigned room number from a reservation, given at least the hotel criteria and reservation identifier. Station identifier is also allowed.
SetDailyPoints	Retrieves a reservation for assigning consumed membership points, given the hotel criteria, confirmation identifier, and list of earned points by date range or start date plus duration.
UpdateInventoryItem	Adds or updates inventory items attached to a reservation, given at least the hotel criteria, confirmation or reservation identifier, item code or item group, and desired quantity and/or date range. Leg confirmation identifier is also allowed. The quantity and date range apply to an item or all items in an item group.
UpdatePackages	Adds or updates a package or package group attached to a reservation, given the quantity. Hotel criteria, confirmation identifier, product code, leg number, and stay date range is also allowed. Details about the package or package group and expected charges by date are provided.

UpgradeReservation	Upgrade Reservation from one room type to another room type.
UpsellReservation	Upsell current reservation to selected offer by user.

Reservation Advanced Service Functions

The following functions are currently provided through the Reservation Advanced Web Service:

Function	Description
AdditionalKeys	Generates additional room keys for an in-house guest, given at least the hotel criteria, reservation identifier or key track, station identifier, and desired number of keys. User identifier, key encoder, and get-key-track indicator are also allowed. If the latter indicator has value true, the key track is returned.
AlternateRooms	Generates a list of alternate room types and rooms for a guest, given the hotel criteria and either reservation identifier or key track. Rate information is also provided.
AssignRoom	Assigns the room number for a reservation, given at least the hotel criteria and reservation identifier. Requested room number and station identifier are also allowed.
CancelCheckIn	Converts the status of a reservation from checked-in to due-in, given at least the hotel criteria, and either reservation identifier or key track. Printer designation and cancellation type summary are also allowed.
CheckIn	Checks a guest in, given at least the hotel criteria and reservation identifier or key track. Credit card information, printer designation, number of keys, key encoder, approval code, getprint-out indicator, and get-key-track indicator are also allowed. If the indicators have value true, the registration is printed and the key track is returned. Check-in information including the invoice number and room number is provided.
CheckOut	Checks a guest out and generates the final bill, given at least the hotel criteria and reservation identifier or key track. Credit card information and printer designation are also allowed.

ExternalPayment	Calls an external payment provider and submits payment data to the provider for a start and finish payment process.
FetchKeyData	Retrieves KeyTrack data attached to the reservation, given the reservation information.
FetchPromotionCode	Retrieves a list of promotion codes available at the hotel.
FetchResPromotionCode	Retrieves promotion codes attached to the reservation.
FetchRoomSetup	Retrieves room setup information, given the room type and room number. Hotel reference information is also allowed. Returns the room type and number, suite type (if applicable), phone number, maximum occupancy, housekeeping section code, smoking preference, and room description, amongst other details about the room is setup.
FetchRoomStatus	Retrieves room status information for the requested room or room type. The response includes the room number and type, next reservation date, room status and the valid dates of the room status, front office status, occupancy condition and housekeeping status, and the service status.
GenerateRegistrationCard	Generates a registration card for the guest during checkin.
GuestMessages	Retrieves the messages not yet flagged as received for a guest, given at least the hotel criteria and reservation identifier or key track. Line length is also allowed. Arriving, departing, and inhouse guests are considered.
Invoice	Retrieves an invoice, given at least the hotel criteria and reservation identifier or key track. Printer designation is also allowed. An itemized bill is provided.
MakePayment	Post a payment, given at least the hotel criteria, reservation identifier or key track, credit card information, and charge amount. Posting date, posting time, short comment, long comment, station identifier, and user identifier are also allowed.

PayRouting	Retrieves lists of routing instructions and payment methods, given the hotel criteria and reservation identifier or key track. Window number, window user, and room number are included. This information is provided only for notification that part of the stay may be handled by different pay windows.
PostCharge	Adds charges to a guest account, given at least the hotel criteria, reservation identifier or key track, and charge amount. Posting date, posting time, short comment, long comment, station identifier, user identifier, account number, and article number are also allowed.
PrintPreCheckOutBill	Generates a bill, given at least the hotel criteria and reservation identifier or key track. Printer designation is also allowed. The bill may either be generated as a PDF file that the kiosk may retrieve over the network and print, or it may be directly printed.
ReleaseRoom	Releases the assigned room number from a reservation, given at least the hotel criteria and reservation identifier. Station identifier is also allowed.
ReservationRequestCode	Retrieves a list of special requests with a reservation, including profile preferences, given the hotel criteria and reservation identifier or key track. The request type, code, source, and description are provided.
SetKeyData	Stores key track data in OPERA, given the hotel reference information, reservation ID, and/or key track data.
SetResPromotionCode	Stores promotional information related to a reservation, given the reservation request information and the promotion code(s).
UpdateMethodOfPayment	Updates the method of payment, given the Hotel Code, Key track or OPERA Reservation Id, Folio View Number, and Credit Card data. From a kiosk, the method of payment can be changed on individual folio windows, not just for the entire reservation.

Security Service Functions

The following functions are currently provided through the Security Web Service:

Function	Description
AuthenticateNRUser	Authenticates a non-registered (no membership profile) user and returns the name identifier, given the user last name, confirmation identifier, and credit card number.
AuthenticateUser	Authenticates a user and returns the name identifier, given the membership number, last name, and password.
CreateAppUser	Creates an application user profile and returns the name identifier, given the profile details, and user group type, as well as a login name, password, and expiration date.
CreateUser	Creates a web user, given the name identifier and a login name and password.
DeleteAppUser	Deletes an application user profile, given the login name and password.
ExtAuthenticateUser	Authenticates a user and returns the login profile, given the login name and password.
FetchAppUser	Retrieves a list of application users, given the login name. Login and name profile details are also returned.
FetchQuestionList	Retrieves a list of predefined secret questions that users can choose from and answer for further authentication.
GeneratePassword	Returns an automatically generated password, given the login name.
LoginAppUser	Authenticates an application user using the login name and password credentials, and returns the security identifier (company name and user group type) and license key.
UpdateAppUser	Modifies an application user profile and returns the name identifier, given the login name, password, and expiration date, as well as the name profile and user group information.

UpdateAppUserPassword	Updates the user's password given the application user's loginName, oldPassword, and newPassword. This allows users to change their passwords without logging into the application. A Success or Error Result is returned.
UpdatePassword	Modifies the login password, given the membership number, last name, old password (if not a new user), and new password.
UpdateQuestion	Modifies the answer to a user's chosen secret question, given the name identifier, pre-defined question identifier, and new answer.
UserAccountLock	Performs a lock or unlock operation on a user account, given the login name and the lock action type.
ValidateQuestion	Validates a user answer to a chosen special question and returns the password, given the membership number, question identifier, and an answer.

Stay History Service Functions

The following functions are currently provided through the Stay History Web Service:

Function	Description
StayHistory	Retrieves a list of past reservations, given the guest name and agent identifiers. A date range is also allowed. Only reservations with status checked-out or cancelled are returned. Considerable detail is provided about the guests, amenities, services, facilities, rates, charges, and communications.

Unit Owners Service Functions

The following functions are currently provided through the Unit Owners Web Service:

Function	Description
AddProfileToContract	Adds a profile to a contract, given the guest name, requesting name, contract, and resort identifiers, and the relationship of the guest to the contract owner.
CreateContract	Create a contract.
DeleteContract	Deletes the contract.
DeleteProfileFromContract	Removes a profile from a contract, given the guest name, requesting name, resort, and resort identifiers.
FetchAuthorizedProfiles	Retrieves a list of contracts, given the name, resort, and/or contract identifiers. Contracts of the owner, co-owners, and family/friends are included. Information about the contract type, unit, effective dates, rate plans, and attached profiles is provided.
FetchContract	Retrieves a contract, given at least the name identifier. The resort and contract identifiers and the requestor role (primaryowner, co-owner, or friends-and-family) are also allowed, and are returned. Information about the contract type, unit, effective dates, rate plans, and attached profiles is provided.
FetchContractDetails	Retrieves details of a contract, given the name and resort identifiers. The contract identifier is also allowed and is returned. Information includes financial arrangements, restrictions, contract documents, and insurance policies in addition to information about the contract type, unit, effective dates, rate plans, and attached profiles.
OwnedUnitAvailability	Retrieves unit availability by contract, given at least the name identifier and stay date range. Contract and resort identifiers as well as a list of guest counts and the room type code are also allowed. For each contract, information includes the contract identifier, type, unit, effective dates, rate plans, and attached profiles, as well as a list of room types with rate details.
OwnerStatementDetails	Retrieves the file names of the owner statement and summary information, given the name and resort identifiers and the stay date range.

UpdateContract	Update contract with new details.
UpdateProfileInContract	Modifies a family/friend guest profile, given the guest name, requestor name, contract, and resort identifiers as well as the relationship of the requestor.

3 Secured Mode

The Secured Mode OWS application parameter, when enabled, determines if User Credentials are required to use the OWS Web Service functions. When the parameter value is set to Y, each Web Service function will follow the OPERA Permissions that are attached to the User Credentials, which are being passed as an argument to the function. If the User Credentials are required to use the Web Service function, the OPERA user permissions will be read to determine whether or not the Web Service function can be used.

The available values for the Secured Mode parameter are Y (Yes) and N (No), which will make the parameter active or inactive, respectively.

The example below shows the Authentication section of a request message with the Authentication complex type for the UserName and UserPassword elements. The Authentication will be included in all request and response messages when the Secured Mode parameter is enabled.

Authentication example

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Header>
  <OGHeader transactionID="15451251" timeStamp="2008-12-08T09:15:28.0368750-05:00"
xmlns="http://webservices.micros.com/og/4.3/Core/">
        <Origin entityID="WEST" systemType="WEB"/>
        <Destination entityID="WEST" systemType="ORS"/>
        <Authentication>
              <UserCredentials>
                    <UserName>USERNAME</UserName>
                    <UserPassword>PASSWORD</UserPassword>
                    <Domain>RESORT</Domain>
              </UserCredentials>
        </Authentication>
  </OGHeader>
</soap:Header>
```

Header

```
<xs:element name="OGHeader" type="tns:OGHeader">
        <xs:annotation>
                        <xs:documentation xml:lang="en">The header values as sent in a SOAP
message for all request to the web service.</xs:documentation>
        </xs:annotation>
</xs:element>
```

Name	Type	Data Type	Use	Comment
OGHeader	element	OGHeader	required	The header values, sent as a SOAP header element in the message.

OGHeader

```
<xs:complexType name="OGHeader">
        <xs:sequence>
            <xs:element name="Origin" type="tns:EndPoint"/>
            <xs:element name="Destination" type="tns:EndPoint"/>
            <xs:element name="Intermediaries" type="tns:EndPointList" minOccurs="0"/>
            <xs:element name="Authentication" minOccurs="0">
                <xs:annotation>
                    <xs:documentation xml:lang="en">Authentication element that is used to identify who
the caller is and if they are allowed to call the web service. This element is optional as it is only used when
the web service is running in secure mode.</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="UserCredentials">
                            <xs:complexType>
```

```
                                        <xs:sequence>
                                                <xs:element name="UserName"
type="xs:string">
                                                <xs:annotation>
                                                        <xs:documentation
xml:lang="en">User Name of the person
requesting to use the application. They can make a booking on behalf of any one
else.</xs:documentation>
                                                </xs:annotation>
                                        </xs:element>
                                                <xs:element name="UserPassword"
type="xs:string">
                                                <xs:annotation>
                                                        <xs:documentation
xml:lang="en">Password that is used to
authenticate the user and is as provided by OPERA.</xs:documentation>
                                                </xs:annotation>
                                        </xs:element>
                                                <xs:element name="Domain"
type="xs:string">
                                                <xs:annotation>
                                                        <xs:documentation
xml:lang="en">Domain identifies the user as
to what Authentication platform they belong to. The default is OPERA and has been added to allow for the
posibility of Federated Authentication at a later stage.</xs:documentation>
                                                </xs:annotation>
                                        </xs:element>
                                                        <xs:element name="SecurityId"
type="xs:string" minOccurs="0"/>
                                        </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                                <xs:element name="Licence" minOccurs="0">
                                        <xs:complexType>
                                        <xs:sequence>
```

```
                                                    <xs:element name="Key" minOccurs="0">

                                                <xs:annotation>

                                                                    <xs:documentation
xml:lang="en">The key will be in the form of
```

```
a GUID and is used to check if the user calling the web service is really authorized to do so or not. The key is issued by OPERA application.</xs:documentation>
                                                </xs:annotation>
                                                <xs:simpleType>
                                                        <xs:restriction base="xs:string">
                                                                <xs:pattern value="\{[\dA-Fa-f]{8}\-[\dA-Fa-f]{4}\-[\dA-Fa-f]{4}\-[\dA-Fa-f]{4}\-[\dA-Fa-f]{12}\}"/>
                                                </xs:restriction>
                                                </xs:simpleType>
                                        </xs:element>
                                        </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                                </xs:sequence>
                        </xs:complexType>
                        </xs:element>
                </xs:sequence>
                                        <xs:attribute name="transactionID" type="xs:string"/>
                                <xs:attribute name="authToken" type="xs:string"/>
                                        <xs:attribute name="timeStamp" type="xs:dateTime"/>
                                        <xs:attribute name="primaryLangID" use="optional" default="E">
                        <xs:annotation>
                                <xs:documentation xml:lang="en">Identifes the primary language preference for the message.  The human language is identified by ISO 639 codes.</xs:documentation>
                        </xs:annotation>
                        <xs:simpleType>
                                <xs:restriction base="xs:string"/>
                        </xs:simpleType>
                </xs:attribute>
</xs:complexType>
```

Name	Type	Data Type	Use	Comment
Origin	element	EndPoint	required	The system that the message originated from (mapped as the channel code).
Destination	element	EndPoint	required	The system that the message will be returned to.
Intermediaries	element	EndPointList	optional	The additional systems that the message will be processed in.
Authentication	element	Authentication	optional	The message authentication details.
transactionID	attribute	string	required	The unique message identification number.
authToken	attribute	string	optional	An authenticated token for the current message.
timeStamp	attribute	string	required	The time stamp for the current message.
primaryLangID	attribute	string	optional	The language that the response message will be displayed in. If not specified, the default language is E (English).

Availability Request with Authentication

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <soap:Header>
  <OGHeader transactionID="256478994" primaryLangID="E" timeStamp="2008-
1010T09:55:16.3618750-05:00" xmlns="http://webservices.micros.com/og/4.3/Core/">
   <Origin entityID="WEST" systemType="WEB" />
   <Destination entityID="TI" systemType="ORS" />
   <Authentication>
    <UserCredentials>
     <UserName>USERNAME</UserName>
```

```
        <UserPassword>PASSWORD</UserPassword>
        <Domain>RESORT</Domain>
       </UserCredentials>
     </Authentication>
   </OGHeader>
  </soap:Header>
  <soap:Body>
   <AvailabilityRequest xmlns:a="http://webservices.micros.com/og/4.3/Availability/"
xmlns:hc="http://webservices.micros.com/og/4.3/HotelCommon/" summaryOnly="true"
xmlns="http://webservices.micros.com/ows/5.1/Availability.wsdl">
    <a:AvailRequestSegment availReqType="Room" numberOfRooms="1"
roomOccupancy="1" totalNumberOfGuests="1" totalNumberOfChildren="0">
     <a:StayDateRange>
      <hc:StartDate>2008-10-10T00:00:00.0000000-05:00</hc:StartDate>
      <hc:EndDate>2008-10-11T00:00:00.0000000-05:00</hc:EndDate>
     </a:StayDateRange>
     <a:RatePlanCandidates>
      <a:RatePlanCandidate ratePlanCode="DBR" />
     </a:RatePlanCandidates>
     <a:HotelSearchCriteria>
      <a:Criterion>
       <a:HotelRef chainCode="WC" hotelCode="SEFIVE" />
      </a:Criterion>
     </a:HotelSearchCriteria>
    </a:AvailRequestSegment>
   </AvailabilityRequest>
  </soap:Body>
</soap:Envelope>
```

Availability Response with Authentication

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
        <soap:Header>
                <OGHeader transactionID="256478994" timeStamp="2008-10-
10T09:55:16.4854640-
                05:00" primaryLangID="E"
xmlns="http://webservices.micros.com/og/4.3/Core/">
                        <Origin entityID="TI" systemType="ORS"/>
                        <Destination entityID="WEST" systemType="WEB"/>
                        <Authentication>
```

```
                                    <UserCredentials>
                                        <UserName>USERNAME</UserName>
                                        <UserPassword>PASSWORD</UserPassword>
                                        <Domain>RESORT</Domain>
                                    </UserCredentials>
                                </Authentication>
                            </OGHeader>

<wsa:Action>http://webservices.micros.com/ows/5.1/Availability.wsdl#AvailabilityRespo
n se</wsa:
                Action>
                        <wsa:MessageID>urn:uuid:e320cf0b-da27-49bf-928e-
6c932da8ea3c</wsa:MessageID>
                        <wsa:RelatesTo>urn:uuid:eb84c110-794f-4f42-
9349338e3463d249</wsa:RelatesTo>

  <wsa:To>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</wsa:To>
                </soap:Header>
                <soap:Body>
                        <AvailabilityResponse
xmlns:hc="http://webservices.micros.com/og/4.3/HotelCommon/"
xmlns:c="http://webservices.micros.com/og/4.3/Common/"
xmlns:a="http://webservices.micros.com/og/4.3/Availability/" summaryOnly="true"
                        xmlns="http://webservices.micros.com/ows/5.1/Availability.wsdl">
                                <Result resultStatusFlag="SUCCESS"/>
                                <AvailResponseSegments>
                                        <a:AvailResponseSegment>
                                                <a:RoomStayList>
                                                        <hc:RoomStay>
                                                                <hc:RatePlans>
                                                        <hc:RatePlan
ratePlanCode="DBR"
                                                                hold="true">
                                                                <hc:RatePlanDescription>


<hc:Text>DBAR1</hc:Text>
                </hc:RatePlanDescription>
                                                                <hc:AdditionalDetails>

                <hc:AdditionalDetail

detailType="MarketingInformation">
<hc:AdditionalDetailDescription>
<hc:Text>Thank you for List Booking
```

```
                                                                    with WEST-
SEFIVE</hc:Text>
</hc:AdditionalDetailDescription>

          </hc:AdditionalDetail>

          <hc:AdditionalDetail

detailType="CommissionPolicy">
<hc:AdditionalDetailDescription>

           <hc:Text>No commission available</hc:Text>

          </hc:AdditionalDetailDescription>

          </hc:AdditionalDetail>

          <hc:AdditionalDetail detailType="TaxInformation">

          <hc:AdditionalDetailDescription>

        <hc:Text>Room tax 15.6 PCT Occupancy Tax 3.6 PCT Transit Tax 1.0 PCT </hc:Text>

        </hc:AdditionalDetailDescription>

         </hc:AdditionalDetail>
                                                                    </hc:AdditionalDetails>

           <hc:CancellationDateTime period="P1DT7H">2008-
1009T01:00:00</hc:CancellationDateTime>
                                                                  </hc:RatePlan>
                                                                  <hc:RatePlan
ratePlanCode="MGM" hold="true">

          <hc:RatePlanDescription>

         <hc:Text>Package Rate MGM Test</hc:Text>

          </hc:RatePlanDescription>
                                                                    <hc:AdditionalDetails>

           <hc:AdditionalDetail detailType="MarketingInformation">
```

```
<hc:AdditionalDetailDescription>
<hc:Text>Thank you for List Booking with WEST-SEFIVE</hc:Text>
</hc:AdditionalDetailDescription>
</hc:AdditionalDetail>
<hc:AdditionalDetail detailType="CommissionPolicy">
<hc:AdditionalDetailDescription>
<hc:Text>No commission available</hc:Text>
</hc:AdditionalDetailDescription>
</hc:AdditionalDetail>
<hc:AdditionalDetail detailType="TaxInformation">
<hc:AdditionalDetailDescription>
<hc:Text>Room tax 15.6 PCT Occupancy Tax 3.6 PCT Transit Tax 1.0 PCT </hc:Text>
</hc:AdditionalDetailDescription>
</hc:AdditionalDetail>
</hc:AdditionalDetails>
<hc:CancellationDateTime period="P1DT7H">2008-1009T01:00:00</hc:CancellationDateTime>
</hc:RatePlan>
<hc:RatePlan ratePlanCode="AMEX" hold="true">
<hc:RatePlanDescription>
<hc:Text>Third Night Free with AMEX Card</hc:Text>
</hc:RatePlanDescription>
<hc:AdditionalDetails>
<hc:AdditionalDetail detailType="MarketingInformation">
<hc:AdditionalDetailDescription>
```

<hc:Text>Thank you for List Booking with WEST-SEFIVE</hc:Text>

</hc:AdditionalDetailDescription>

</hc:AdditionalDetail>

<hc:AdditionalDetail detailType="CommissionPolicy">

<hc:AdditionalDetailDescription>

<hc:Text>Commission policy is 10 PERCENT</hc:Text>

</hc:AdditionalDetailDescription>

</hc:AdditionalDetail>

</hc:AdditionalDetails>

<hc:CancellationDateTime period="P327906DT8H">1111-0101T00:00:00</hc:CancellationDateTime>

</hc:RatePlan>

</hc:RatePlans>

<hc:RoomTypes> <hc:RoomType roomTypeCode="KNG" numberOfUnits="93">

<hc:RoomTypeDescription>

<hc:Text>K - King Room</hc:Text>

</hc:RoomTypeDescription>

</hc:RoomType>

</hc:RoomTypes>

<hc:RoomRates>

<hc:RoomRate roomTypeCode="KNG" ratePlanCode="DBR">

<hc:Rates>

<hc:Rate rateOccurrence="DAILY" rateChangeIndicator="false">

<hc:Base currencyCode="USD">200</hc:Base>

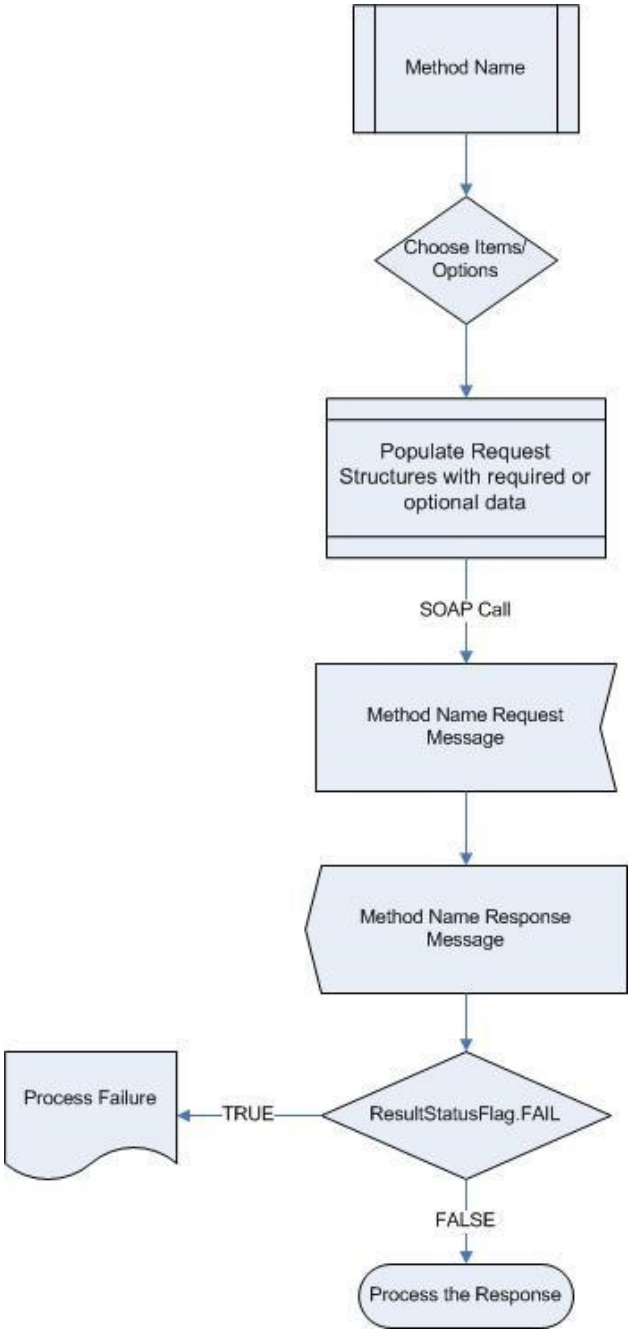
</hc:Rate>

```
                                                    </hc:Rates>
                                                    <hc:Total
currencyCode="USD">200</hc:Total>
                                                </hc:RoomRate>
                                                <hc:RoomRate
roomTypeCode="KNG" ratePlanCode="MGM">
                                                    <hc:Rates>
                                                        <hc:Rate
rateOccurrence="DAILY" rateChangeIndicator="false">
                <hc:Base currencyCode="USD">153</hc:Base>
                                                        </hc:Rate>
                                                    </hc:Rates>
                                                    <hc:Total
currencyCode="USD">153</hc:Total>
                                                </hc:RoomRate>
                                                <hc:RoomRate
roomTypeCode="KNG" ratePlanCode="AMEX">
                                                    <hc:Rates>
                                                        <hc:Rate
rateOccurrence="DAILY" rateChangeIndicator="false">
                <hc:Base currencyCode="USD">117.55</hc:Base>
                                                        </hc:Rate>
                                                    </hc:Rates>
                                                    <hc:Total
currencyCode="USD">117.55</hc:Total>
                                                </hc:RoomRate>
                                            </hc:RoomRates>
                                        <hc:HotelReference chainCode="WC"
hotelCode="SEFIVE">Opera Demo Hotel, Medium</hc:HotelReference>
                                            <hc:HotelContact>
                                                <hc:Addresses>
                                                    <hc:Address>
            <c:AddressLine>1415 Fifth Ave.</c:AddressLine>
                <c:cityName>Naples</c:cityName>
                <c:countryCode>US</c:countryCode>
                                                    </hc:Address>
```

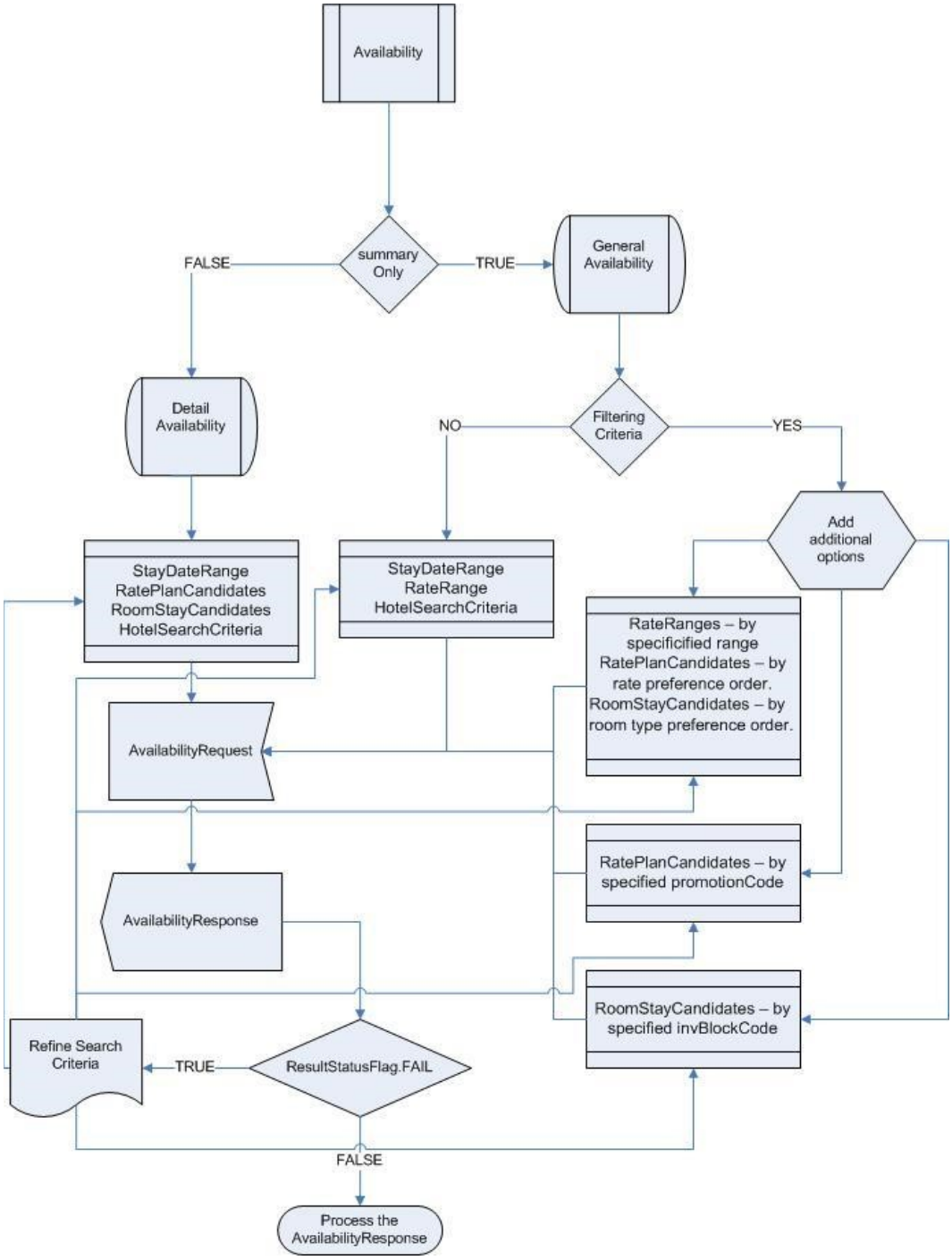
```
                                                    </hc:Addresses>
                                                    <hc:ContactEmails>

                <hc:ContactEmail>5thavesales@hotels.com</hc:ContactEmail>
                                                    </hc:ContactEmails>
                                                    <hc:ContactPhones>
                                                        <hc:Phone
phoneType="FAX">

            <c:PhoneNumber>9543764555</c:PhoneNumber>
                                                        </hc:Phone>
                                                        <hc:Phone
                                        phoneType="VOICE">

            <c:PhoneNumber>2397895678</c:PhoneNumber>
                                                        </hc:Phone>
                                                    </hc:ContactPhones>
                                                    <hc:HotelInformation>
                                                        <hc:HotelInfo
hotelInfoType="CHECKININFO">
                                                            <c:Text>

                <c:TextElement>08:00</c:TextElement>
                                                            </c:Text>
                                                        </hc:HotelInfo>
                                                        <hc:HotelInfo
hotelInfoType="CHECKOUTINFO">
                                                            <c:Text>

                <c:TextElement>15:00</c:TextElement>
                                                            </c:Text>
                                                        </hc:HotelInfo>
                                                    </hc:HotelInformation>
                                                </hc:HotelContact>
                                            </hc:RoomStay>
                                        </a:RoomStayList>
                                    </a:AvailResponseSegment>
                                </AvailResponseSegments>
                            </AvailabilityResponse>
                        </soap:Body>
</soap:Envelope>
```

4 Logical Workflow Diagrams

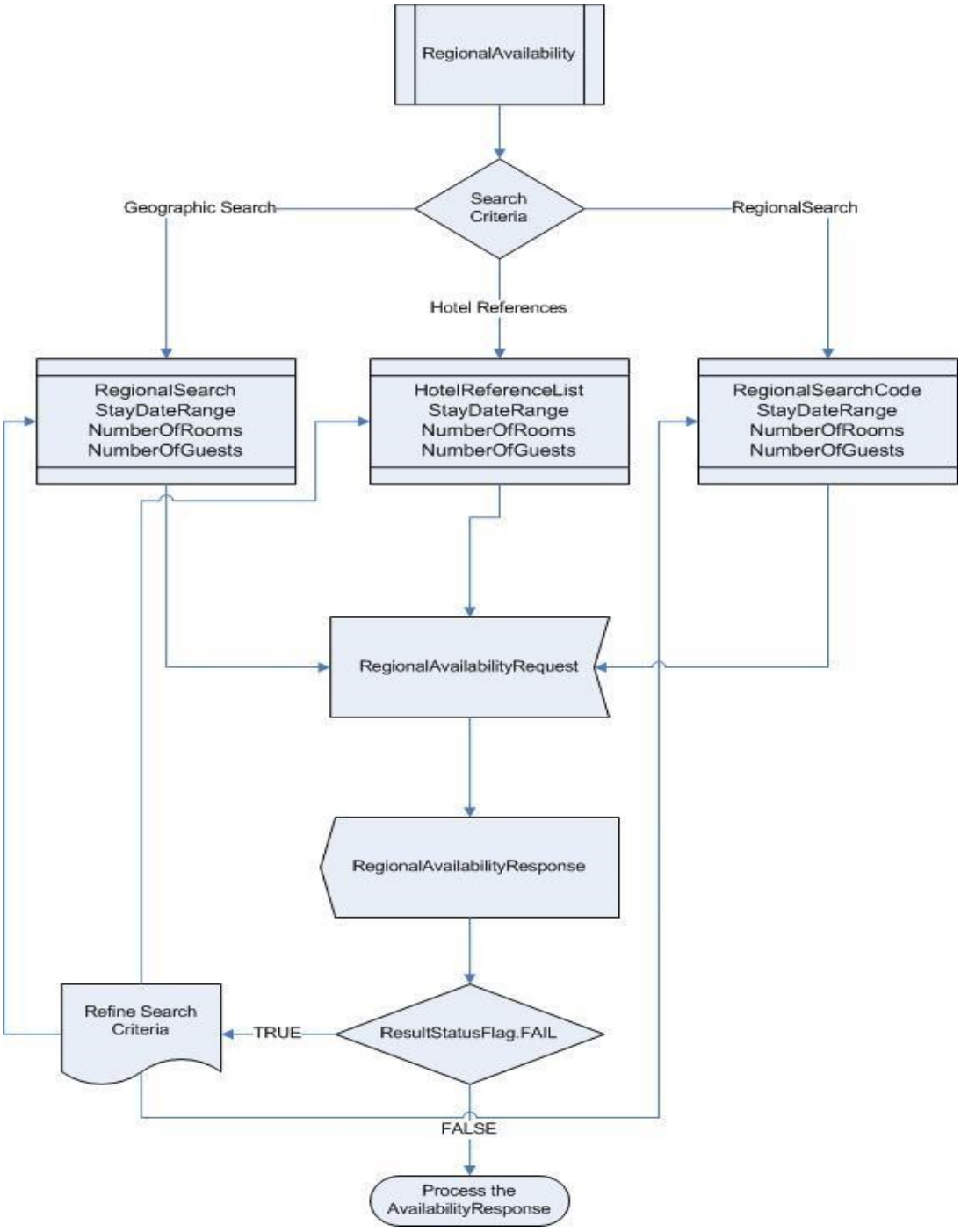
Common Logical flowchart for all services:



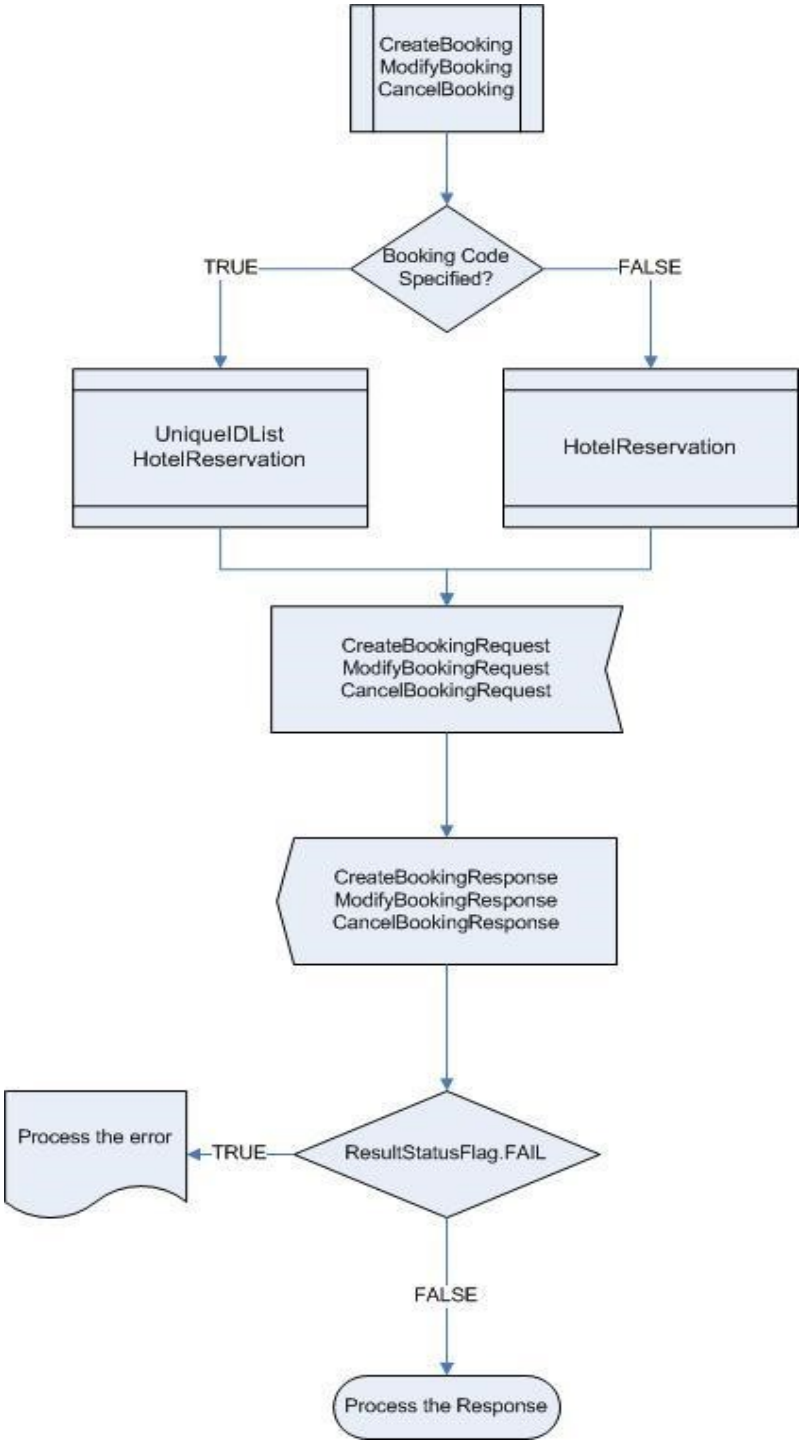
Availability Service - Availability method flowchart:



Availability Service – RegionalAvailability method flowchart:



Reservation Service – CreateBooking/ModifyBooking/CancelBooking method flowchart:



5 The SOAP Header and OPERA Channels

OPERA allows companies that use Global Distribution System (GDS) or Alternative Distribution Systems (ADS), such as central reservation systems, travel agents, and Web sites, access to read and write to the database. For example, a travel agent can use GDS to inquire about a property's availability, and then make a reservation that is stored in the database. To have channels work properly in an OPERA environment, you need to:

- Create and configure channels using OPERA Reservation System (ORS) in configuration mode. The channel you create in OPERA is associated with the GDS/ADS host, and can be for the hotel's Web site, one of the four GDS such as Amadeus, or ADS such as Expedia.
- Map conversion codes used in OPERA to the codes used on the GDS/ADS host. For example, if you have a corporate rate for a global company, you need to map this rate code to the GDS/ADS access code equivalent. This means that all travel agents in the world can see the same corporate rate and property codes, regardless of which channel or system they are using.

For more information, refer to OPERA Channel Management in OPERA 5 online help.

Header

Channel configurations affect the setup of the SOAP header. In OWS, a SOAP header is represented as "OGHeader". Below is an example of the code:

System.Xml.Serialization.XmlTypeAttribute(Namespace="http://webservices.micros.com/og/4.3/C ore/")]

[System.Xml.Serialization.XmlRootAttribute(Namespace="http://webservices.micros.com/og/4.3/
Core/", IsNullable=false)] public class OGHeader :

System.Web.Services.Protocols.SoapHeader { public

EndPoint Origin; public EndPoint Destination;

[System.Xml.Serialization.XmlArrayItemAttribute(IsNullab

le=false)] public EndPoint[] Intermediaries; public

OGHeaderAuthentication Authentication;

```
[System.Xml.Serialization.XmlAttributeAttribute()]   public

string transactionID;

  [System.Xml.Serialization.XmlAttributeAttribute()]
  public string authToken;


[System.Xml.Serialization.XmlAttributeAttrib

ute()]   public System.DateTime timeStamp;

[System.Xml.Serialization.XmlIgnoreAttribute

()]   public bool timeStampSpecified; }
```

Origin, Destination and Intermediaries

Origin, destination and intermediaries must have channel and channel type set up. The origin defines where the request will be received, and the destination is where the request originated from. In case there are intermediaries in-between, an origin and destination list can be added. For example:

```
      OGHeader oHead = new OGHeader();
      …
      //Origin Endpoints
 oHead.Origin   =   new   EndPoint();
oHead.Origin.entityID     =     “AA”;
oHead.Origin.systemType    =    “GDS”;
//Destination               Endpoints
oHead.Destination  =  new  EndPoint();
oHead.Destination.entityID   =   “TI”;
oHead.Destination.systemType  = “ORS”;
      //Intermediates Endpoints

      EndPoint ep1 = new EndPoint();
ep1.systemType = oHead.Origin.entityID;
```

Transaction ID

The transaction ID is a six-digit random number.

The Authorization Token

The authorization token is unique ID similar to the global unique identifier, such as

{82DA3509-6DF5-4302-AE17-B1417068F351}.

Time Stamp

The time stamp is the time the request is being made, and timeStampSpecified is a Boolean flag which must be true in case you specify a timestamp; otherwise, it is not turned on.

Authentication

Authentication is where you specified the user credentials that will be used for the session, such as:

```
//Authentication Endpoints

oHead.Authentication = new

OGHeaderAuthentication();

oHead.Authentication.UserCredentials = new

OGHeaderAuthenticationUserCredentials();

oHead.Authentication.UserCredentials.UserName = “IBMBOOKER”;

oHead.Authentication.UserCredentials.UserPassword = “IBMBOOKER”;

oHead.Authentication.UserCredentials.Domain = “IBMDOMAIN”;
```

6 Payment and Guarantee

Each hotel has its own available and predefined methods for payment and guarantees, as well as accepted credit card types. To obtain information about the available and predefined methods and accepted credit card types for a specific hotel, use the **QueryHotelInformation** method from the information service.

The tables below describe the entire set of payment methods, guarantee methods, and credit card types.

Valid abbreviations for a payment are:

Abbreviation	**Description**
CA	Cash
VO	Voucher
IN	Invoice
PR	Prepayment
CC	Credit Card

Valid abbreviations for guarantee method are:

Abbreviation	**Description**
GT	Guaranteed by travel agent
GX	Guaranteed by credit card
CO	Guaranteed by company name
GC	Guaranteed by corporate ID
GP	Guaranteed by deposit
DX	Guaranteed by credit card deposit
XX	Guaranteed to release time

Valid abbreviations for credit cards accepted are:

Abbreviation	**Description**
AB	Bank card
AX	American Express
BB	Barclay card
BF	Carte Bleu
CB	Credit Blanche
DC	Diner's card
DI	Discover card
EC	Eurocard
IL	En route
JC	Japanese Credit Bureau
MC	Master Card
VI	Visa
XS	Access Card

7 Restrictions

- It is the user's responsibility to control that unique NameIDs and TransactionIDs are used.
- If OWS is used via the Internet, a security check is needed for the OWS server that the application is run on.
- Up to four people can be booked for one room in each request.
- Restrictions can be setup from Channel Management.

8 Availability Limitations

Tuning Note - When utilizing both General and Regional Availability Calls Simultaneously for Websites displaying availability with a single click:

In most cases, regional availability, general availability response time is less than a second from OPERA to the website. However calling the web service in and in-appropriate way can cause degradation in response time and performance. An example of this can be seen when a User is searching availability in Dallas, this requires the website to make a call to regional availability requesting Dallas as the city, and the response returns with all available properties in Dallas, with minimum and maximum rate amounts for each property. Once the list of properties are returned to website, the user picks one property at a time and makes a call to general availability for that property.

Some websites make a call to general availability for all properties returned by regional availability without the user selecting a specific property, and this increases time response. This process is known as "stacking" calls. We do not recommend this procedure as a best practice. If the client does decide to implement this process, they should be aware that the below formula needs to be used to gain an accurate view of the potential impact this may have on the guest experience on the web.

Stacking both General and Regional availability calls back-to-back, can be calculated by using the following equation:

Rt = ((2 x Nm) x (5 x Nr) x (2xNd)) x Np + ((5 x Ng) x Nm x (5 x Np))

Where: Rt is total response time in milliseconds

Nm – number of rooms

Nr – number of rates

Nd – number of days
Np – number of properties covered by the General/Regional availability request(s)

Ng – number of guests

9 Security

A Secure Sockets Layer (SSL) is a set of cryptographic technologies that provides authentication, confidentiality, and data integrity. SSL is most commonly used between Web browsers and Web servers to create a secure communication channel. It can also be used to secure communications between client applications and Web Services.

All data over a session must be protected from unauthorized parties. All servers used by OPERA Web Suite (OWS) implementers must maintain access control in order to prevent unauthorized use of the system.

Implementers must support SSL and must accept self-signed certificates or those signed by a recognized certificate authority (VerySign Inc., Thawte Inc.).

Even experienced Web Services developers often get caught by the default secure SSL configuration. Refer to additional information on how to install an SSL certificate.

10 OWS Glossary

SOAP	Simple Object Access Protocol (SOAP) is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. SOAP uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation specific semantics.
XML	Extensible Markup Language (XML) describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of Standard Generalized Markup Language (SGML) [ISO 8879]. XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure.
URI	Uniform Resource Identifier (URI) is the generic term for all types of names and addresses that refers to objects on the World Wide Web.
XSD	XML Schema Definition language (XSD) is the current standard schema language for all XML documents and specifies how to formally describe the elements in an XML document.

WSDL	Web Services Description Language (WSDL) is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate, however, the only bindings described are in conjunction with SOAP 1.1, HTTP GET/POST, and MIME.
Java	Java is an object-oriented language similar to C++, but simplified to eliminate language features that cause common programming errors. Java source code files (files with a .java extension) are compiled into a format called byte code (files with a .class extension), which can then be executed by a Java interpreter. Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (VMs), exist for most operating systems. Byte code can also be converted directly into machine language instructions by a just-in-time compiler (JIT).

.NET	A Microsoft operating system platform (similar to Java) that incorporates applications and common languages such as C#, VB.NET, a suite of tools and services and a change in the infrastructure of the company's Web strategy. There are four main principles of .NET from the perspective of the user: • It erases the boundaries between applications and the Internet. Instead of interacting with an application or a single Web site, .NET will connect the user to an array of computers and services that will exchange and combine objects and data. • Software will be rented as a hosted service over the Internet instead of purchased on a store shelf. Essentially, the Internet houses all of the applications and data. • Users will have access to their information on the Internet from any device, anytime, anywhere. • There will be new ways to interact with application data, such as speech and handwriting recognition.
Perl	Practical Extraction and Report Language (Perl) is a programming language, especially designed for processing text. Because of its strong text processing abilities, Perl has become one of the most popular languages for writing executable scripts. Perl is an interpretive language, which makes it easy to build and test simple programs.
PHP	PHP Hypertext Preprocessor is a server-side, HTML-embedded scripting language used to create dynamic Web pages. In an HTML document, PHP script (similar syntax to that of Perl or C) is enclosed within special tags, similar to Active Server Pages and Cold Fusion.

ASP	Active Server Pages (ASP) provides a server-side scripting environment that can be used to create and run dynamic, interactive Web server applications. ASP combines HTML pages, script commands, and Component Object Model (COM) components to create interactive Web pages, or powerful web-based applications, written in Windows scripting languages such as VBScript and Jscript.

11 OWS Appendix A: Status Codes

The status code's description can be setup from Channel Management.

ERROR CODE	DESCRIPTION
INVALID_OR_MISSING_SEARCH_TYPE	The search key word is invalid for the chain or property.
PROPERTY_NOT_FOUND	The property was not found.
NO_PROPERTY	The property does not exist.
NUMBER_OF_MIN_ADV_DAYS_REQUIRED_BOOK	The minimum number of advance days required to book.
NUMBER_OF_MAX_ADV_DAYS_REQUIRED_BOOK	The maximum number of advance day required to book.
INVALID_ALLOTMENT_CODE	The allotment code is not valid for the property.
INVENTORY_NOT_AVAILABLE_BLOCK_CODE	The inventory for block code is not available for the property.
ROOM_RESTRICTED_BLOCK_CODE	The rooms are restricted for a block code.
INVALID_PROMOTION_CODE	The promotion code is invalid for the property.
INVALID_RATE_CODE_FOR_PROMOTION_CODE	PROMOTION CODE IS NOT VALID FOR RATECODE
OWSNAME-000002	The name ID is null.

OWSNAME-000003	The first name is null.
OWSNAME-000004	The last name is null.
OWSNAME-000005	The name type is null or invalid.
OWSNAME-000006	The native fields are null.
OWSADD-000001	Invalid Address ID.
OWSADD-000002	Invalid Address Type.

ERROR CODE	DESCRIPTION
OWSADD-000003	The country was not specified.
OWSADD-000004	Invalid country.
OWSADD-000005	Invalid city.
OWSPHONE-000001	Invalid Phone ID.
OWSPHONE-000002	The phone number cannot exceed 2000 characters.
OWSPHONE-000003	Invalid phone type.
OWSPHONE-000004	Phone number is null.
OWSEMAIL-000001	Invalid Email ID.
OWSEMAIL-000002	Email ID is greater than 2000 characters.

OWSEMAIL-000003	Invalid Email address.
OWSEMAIL-000004	Email address is null.
OWSPASS-000001	The passport number is null.
OWSPASS-000002	The passport number is greater than 40 characters.
OWSPASS-000003	The passport issue date is null.
OWSPASS-000004	The passport place of issue is null.
OWSPASS-000005	The passport place of issue is greater than 80 characters.
OWSGEN-000001	The message ID is null.
OWSCC-000001	The credit card type is not specified.
OWSCC-000002	Invalid credit card type.
OWSCC-000003	The credit card number is not specified.
OWSCC-000004	Invalid credit card number.
OWSCC-000005	Invalid credit card ID.

ERROR CODE	DESCRIPTION
OWSCC-000006	The credit card name is not specified.
OWSCC-000007	Invalid credit card type.
OWSCC-000008	The expiration date is null.
OWSCC-000009	Invalid expiration date.

OWSCC-000010	The credit card with this type and number already exists.
OWSCC-000011	The credit card type cannot be updated.
OWSMEM-000001	The membership type is null.
OWSMEM-000002	The membership card is null.
OWSMEM-000003	The name on the card is null.
OWSMEM-000004	Duplicate membership.
OWSMEM-000005	Invalid membership ID.
OWSMEM-000006	The expiration date is null.
OWSMEM-000007	The membership type cannot be updated.
OWSPRE-000001	The preference type is null.
OWSPRE-000002	The preference value is null.
OWSPRE-000003	Invalid preference type.
OWSPRE-000004	Invalid preference value.
OWSPRE-000005	The record does not exist.
OWSCOMM-000001	Invalid comment ID.
OWSCOMM-000002	The comment title and comment is null.
OWSCOMM-000003	The comment ID is null.
MISSING_CONFIRMATION_NO	No confirmation number was found.
TOO_LATE_TO_MODIFY_RESERVATION	Changing this reservation is not allowed.

ERROR CODE	DESCRIPTION
INVALID_RATE_CODE_FOR_ALLOTMENT_CODE	The rate code is not valid for this channel.
OWSNAME-000001	Invalid name ID.

OWSSH-000001	No stay history was found for the name ID in the date range provided.
OWSSS-000001	The login name is null.
OWSSS-000002	Invalid login name.
OWSSS-000003	The password is null.
OWSSS-000004	Invalid password.
OWSSS-000005	The guest's last name is null.
OWSSS-000006	The confirmation number is null.
OWSSS-000007	The credit card number is null.
OWSSS-000008	The booking does not exist for the given combination of last name, confirmation number, and credit card number.
OWSSS-000009	The user already exists.
OWSSS-000010	An error occurred when inserting the record.
OWSSS-000011	The record was not updated.
OWSSS-000012	The secret question is null.
OWSSS-000013	Invalid secret question code.
OWSSS-000014	The secret answer is null.
OWSSS-000015	The secret question/answer was not updated.

OWSSS-000016	Incorrect secret answer.
OWSSS-000017	The name ID was already registered.
GUEST_NAME_NOT_ENTERED	The guest's name was not entered, or an invalid profile ID was provided.

ERROR CODE	DESCRIPTION
OWSSS-000018	Duplicate record.
OWSSS-000019	The new password is null.
OWSSS-000020	The old and new passwords are identical.
OWSSS-000021	The name ID is not registered
MISSING_MARKETCODE_OR_TRXCODE_FOR_RATECODE	The market code or transaction code is not setup for the rate code on the rate setup.
RESTRICTION_ON_CANCELORMODIFY	The user is not allowed to modify or cancel the reservation.
ERR_INS_RES_HEADER	Not used.
FREQUENT_GUEST_NUMBER_REQUIRED	A guest membership number is required.
ROLLAWAY_NOT_AVAILABLE	The requested rollaway product is not available.
CRIB_NOT_AVAILABLE	The requested crib product is not available.
INVALID_CREDIT_CARD_NUMBER	Invalid credit card number.
TOO_LATE_TO_BOOK	It is too late to book, modify, or cancel the reservation.
INVALID_CREDIT_CARD	The credit card information is not valid.
INVALID_GUARANTEE_INFO	The guarantee information is not valid.
INVALID_GUARANTEE	The guarantee code is not valid.
GUARANTEE_REQUIRED	The guarantee is required.

INVALID_CORPORATE_GUARANTEE	The corporate guarantee code is not valid.
GUARANTEE_NOT_ACCEPTED	The guarantee provided is not accepted for the reservation.
INVALID_TRAVEL_AGENT_GUARANTEE	The travel agent guarantee code provided is not valid.
INVALID_CREDIT_CARD_EXPIRATION	The credit card expiration date is not valid.

ERROR CODE	DESCRIPTION
NUMBER_ROOMS_EXCEEDS_LIMIT	The requested number of rooms exceeds the number of rooms the system allows you to book.
INVALID_PROPERTY	Invalid property code.
INVALID_RATE_CODE	Invalid rate code.
RATE_CODE_UNAVAILABLE	The rate code is not available.
ROOM_UNAVAILABLE	The room type is not available.
INVALID_ROOM_CATEGORY	Invalid room category or room type.
ROOM_INV_NOT_AVAILABLE	The requested room category is not available.
ROOM_RESTRICTED	The room is restricted.
PROPERTY_NOT_AVAILABLE	The requested property is not available.
PRIOR_STAY	The requested stay is prior to the business date.
INVALID_ACTION_CODE	The action code is invalid.
TOO_LATE_TO_CANCEL	It is too late to cancel the reservation.
INVALID_CARRIER	The carrier code is invalid.
BOOKING_PREVIOUSLY_CANCELLED	The booking was previously cancelled.
CORPORATE_NUMBER_INVALID	The corporate number is not valid.
CORPORATE_NUMBER_REQUIRED	The corporate number is required.
CONFIRM_NO_MUST_EXIST	The confirmation number is invalid or does not exist.
BOOKING_MUST_EXIST	The booking does not exist.

FREQUENT_GUEST_NUMBER_INVALID	The frequent guest or membership number is invalid.
INVALID_CURRENCY_CODE	The currency code is invalid. CURRENCY CODE IS INVALID
DAYS_STAY_MISMATCH	The number of nights and the arrival/departure days do not match.
TOO_MANY_EXTRAS	Not used.

ERROR CODE	DESCRIPTION
NUMBER_NIGHTS_EXCEEDS_LIMIT	The requested number of nights exceeds the number of nights the system allows you to book.
AMADEUS_MULTINAME_RATE	Not used.
CANCEL_UPDATE_HD_ERROR	CANCEL_UPDATE_HD_ERROR
GENERAL_UPD_FAILURE	There are some critical database errors. Please contact the networks administrator.
GUEST_NAME_CANNOT_BE_DELETED	You are not allowed to delete the guest name.
GUEST_NAME_CANNOT_BE_MODIFIED	You are not allowed to make any changes to the guest name.
INVALID_CHAINCODE	The chain code is invalid.
RATE_PLAN_MISSING	The rate plan code is missing.
MULTI_LEG_ERROR	The multi-leg reservation cannot be modified through the GDS.
NUMBER_ADULTS_EXCEEDS_LIMIT	The requested number of adults exceeds the number the system allows you to book.

NUMBER_OF_NIGHTS_CTA_RESTRICTION	A close to arrival restriction exists.
NUMBER_OF_NIGHTS_MLOS_RESTRICTION	A minimum length of stay restriction exists.
OUT_DATE_NOT_IN_INVENTORY	A close to departure restriction exists.
PREV_SEGMENT_FAILED	The previous segment failed.
RATE_CODE_NOT_AVAILABLE	The rate code is not available.
RATE_INVALID	The rate code is invalid.
MAX_OCCUPANCY	The requested occupancy is more than the system allows you to book for the room type.
CTA_RESTRICTION_ON_IN_DATE	A close to arrival restriction exists.
MAX_RATE_INVALID	The maximum rate is invalid.

ERROR CODE	DESCRIPTION
MIN_RATE_INVALID	The minimum rate is invalid.
MISSING_CHAIN_CODE	The chain code is missing.
MISSING_START_DATE	The start date is missing.
MISSING_END_DATE	The end date is missing.
PROPERTY_RESTRICTED	The property is restricted.
REQUEST_NOT_PROCESSED	Unable to process the request.
NUMBER_ROOMS_MISSING	The number of rooms is missing.
INVALID_CITY_CODE	Invalid city code.
CANCELLATION_CONFIRMED_PENALTIES_APPLIED	The cancellation was confirmed and the penalties were applied.

UNABLE_TO_CANCEL_ARRIVAL_DATE_HAS_PASSED	Unable to cancel. The arrival date has passed.
BOOKING_NOT_FOUND	The booking was not found.
END_DATE_INVALID	The end date is invalid.
START_DATE_INVALID	The start date is invalid.
MISSING_ROOM_TYPE	The room type is missing.
MISSING_PROP_ID	The property ID is missing.
MISSING_ROC	The ROC is missing.
INVALIDGDS	Invalid GDS.
TOOMANYRATEPLANS	Too many rate plans.
SYSTEM_ERROR	A system error has occurred.
RESTRICTION_ON_CANCEL_OR_MODIFY_ CONTACT_HOTEL	There is a restriction placed to cancel or modify. Please contact the hotel.
SESSION_NOT_UPDATED	The reservation is in a still session.
ERROR CODE	DESCRIPTION
OTHER_SESSION	The reservation is in a still session.
DEPOSIT_REQUIRED	A deposit required to complete the booking.
OWSCON-000001	No Record was found for the combination of last name, confirmation number, and/or credit card number.
OWSNAME-000007	Invalid membership number/name ID.
OWSNAME-000008	The membership number/name ID is null.
OWSCEMAIL-000001	The confirmation letter is null.
OWSCEMAIL-000002	The email address is null.
OWSCEMAIL-000003	The confirmation number is null.
OWSBR-000001	The required send method is not configured for this property.

OWSBR-000002	Email ID is not found for this email address.
OWSBR-000003	Failed to get the script.
OWSBR-000004	Failed to get the Html text.
OWSBR-000005	An error occurred in sending the email.
OWSBR-000006	An error occurred in sending the email.
INVALID_COMPANY_ID_OR_SECURITY_ID	The company ID or secure ID is not valid or is null.
ALLOCATION_OR_NEGOTIATED_RATE_NOTSETUP_FOR_COMPANY	The allocation or negotiated rates are not set up for this company. Please correct the setup.
OWSCEMAIL-000004	Invalid confirmation number or the reservation was not found.

12 OPERA Xchange Interface (OXI)

Introduction

This document explains the communication mechanisms supported by OPERA Xchange Interface (OXI). Since the details are technical in nature, target audience of this document is developers who are familiar with the fundamentals of the communication mechanisms explained.

OXI supports following communication mechanisms. Details of each mechanism are provided later.

- HTTP/HTTPS
- FTP
- File System

When OXI is implemented as a 2-way interface, the most fundamental operation it needs to do is receiving data to process from an external system and sending data to the external system for processing.

Synchronous or Asynchronous

Even though some of these communication mechanisms such as HTTP support synchronous communication, OXI processing is always asynchronous.

When OXI receives data from an external system, that data may not be processed immediately. It is queued for processing and a synchronous response is provided to notify the status of the receipt of the message whether it has been successfully received or an error occurred while receiving/queuing the message – following the standards defined for that particular communication mechanism, if the communication mechanism supports a synchronous response. For example, 200 OK/400 Bad Request in case of http communication. An asynchronous response conforms to the result XML schema is generated for the external system stating the completion status of the processing and any other relevant information later, once the queued data is processed.

Similarly, when OXI delivers messages to external systems, it expects only an acknowledgement of the receipt of the message in the synchronous response, if applicable. Processing status and any relevant detail should be notified as an asynchronous message that conforms to the result XML schema.

Push or Pull

Communication mechanisms that use the client/server terminology such as HTTP and FTP can be implemented in different configurations. Since there are two parties involved in the communication – in our case OXI and the external system – theoretically either can function as the server and the other party can function as client. OXI does not support all such combinations of configurations due to various reasons. Supported configurations,

where applicable, are explained while discussing the corresponding communication mechanisms later in this document.

HTTP/HTTPS

OXI supports the four possible client server configurations while communicating using HTTP/HTTPS. Following details apply to both HTTP and HTTPS. To use HTTPS, server has to be configured accordingly.

Important: HTTPS functionality when OXI functions as HTTP Client was tested and verified only using Apache HTTP server 1.3 with Verisign's SSL certificate.

Configuration Options

Push/Push

Both systems push messages to other system when messages are available. External system sends messages for OXI to OPERA's HTTP server and OXI sends messages to external system's HTTP server.

Pros	*Cons*
No polling is involved to check the availability of messages. Reduces network activity.	Both systems have to maintain HTTP servers.

Push/Pull

- External system pushes messages to Opera when messages are available and polls OPERA's HTTP server for availability of messages (or)
- OPERA pushes messages to external system when messages are available and polls external system's HTTP server for availability of messages.

Pros	*Cons*
Only one side need to have the HTTP server.	To receive messages, one system will have to poll the other system's HTTP server for available messages at frequent intervals. This increases network activity.

Pull/Pull

Both systems poll the other system for messages and pull them when available. This configuration is included for the completion of the discussion and should not be implemented as this does not have any pros and have all cons of the previous two options.

Recommendations

Either first or second option should be selected after weighing the pros and cons of both options. Third option is not recommended at all as it does not seem to have any pros and has combined cons of the other two options.

OXI as Client: Receive Data from External System

An HTTP GET operation will be performed to retrieve available messages for the property from the external system. If message is available for the property, external system should return 200 OK and should send XML message in the response stream. If there is no message available for the property, external system should set the value of Content-Length in the response header to 0 and return 200 OK/204 No Content. At the time of this writing, OXI relies on the 0 Content-Length to recognize that there is no message.

HTTP Operation :

GET

URL Parameters

Notes : Parameter names will be in camel case as shown below and values will be in upper case.

Name	***Description***
propertyName	Resort code used by the external system to identify the property. This will be the same as external resort set in OXI's configuration.

Expected HTTP Response

Result	***Status Code***	***Content-Type***	***Content-Length***	***Content***
Message available	200 OK	text/xml (UTF-8)	not used	XML message
Message not available	204 No Content/ 200 OK	not applicable	0	None
Error	>= 400	text/plain	not used	Error message

Mandatory Processing Instruction

To parse the XML message received from the external system, OXI needs to know some basic information about the message. OXI expects a processing instruction in the XML Message as follows immediately after the XML version information, to derive this information from.

<?Label *propertyName*|*messageType*|*transactionId*|*status*?>

Tags shown in green italic above must be replaced with actual values. Details of these tags follow:

Name	***Description***
propertyName	Resort code used by the external system to identify the property. This should be the same as external resort set in OXI's configuration.
messageType	The type of XML message sent by the external system. The following values are valid: ALLOTMENT : Allotment INVENTORY : Inventory PROFILE : Profile RATE : Rate RAVL : Rate restriction RAVR : Rate restriction (room type) RESERVATION : Reservation RESULT : Result
transactionId	For new messages generated by the external system, external system's numeric transaction identifier that uniquely identifies this particular message. It must be between 1 and 999999999 inclusive. For RESULT messages generated in response to OXI's message, message id of the original message sent by OXI – It is used to update the status of the original message sent by OXI accordingly.
status	Reserved. Must be SUCCESS always.

Sample:

<?xml version = '1.0'?>
<?Label

SLBK|PROFILE|249|SUCCESS?

> Sample URL:

`http://oxihub.ft.micros.com:5868/servlets/ORSInterface?propertyName=SLBK`

http://oxihub.ft.micros.com:5868/servlets/ORSInterface :
Configurable in OXI.
SLBK : Will be replaced by OXI using actual value.

OXI as Client: Send Data to External System

HTTP Operation : POST
Content-type : text/xml
Character-encoding : UTF-8

URL Parameters
Notes : Parameter names will be in camel case as shown below and values will be in upper case.

Name	***Description***
propertyName	Resort code used by the external system to identify the property. This will be the same as external resort set in OXI's configuration.
messageType	The type of the XML message sent by OXI. The following values are valid: ALLOTMENT : Allotment HURDLE : Hurdle INVENTORY : Inventory
	INVENTORYQUERY : Inventory request INVENTORYSUMMARY : Inventory summary MESSAGEREQUEST : Message request PACKAGES : Package PROFILE : Profile RATE : Rate RAVL : Rate restriction RAVR : Rate restriction (room type) RESERVATION : Reservation RESULT : Result RTAV : Inventory snapshot SCHEMAVERSION : XML schema version

	STAY	: Guest stay history
transactionId	For new messages generated by OXI, OXI's numeric transaction identifier that uniquely identifies this particular message. It will be between 1 and 999999999 inclusive. For RESULT messages generated in response to external systems' message, message id of the original message sent by the external system.	
status	NEW	: New non-result message.
	SUCCESS	: Only for result messages. OXI processed original message received from the external system was successfully.
	FAILED	: Only for result messages. OXI could not process original message received from the external system.
	DELETE	: Only for profile messages. User deleted the profile in Opera.

Expected HTTP Response

Result	***Status Code***	***Content-Type***	***Content-Length***	***Content***
Message accepted	200 OK	not applicable	not used	Not used
Error encountered	>= 400	text/plain	not used	Error message. OXI will resend the message after a delay.

Sample URL:

```
http://oxihub.ft.micros.com:5868/servlets/ORSInterface?propertyNam
e=SLBK&messageType=PROFIL E&transactionId=208&status=SUCCESS
```

http://oxihub.ft.micros.com:5868/servlets/ORSInterface :
Configurable in OXI.
SLBK, PROFILE, 208, SUCCESS : Will be replaced by OXI using actual values.

OXI as Server: Send Data to OXI

HTTP Operation : POST
Content-type : text/xml
Character-encoding : UTF-8
URL :
http://*OperaHTTPServer*:*Port*/*Operajserv*/*OXIServlets*/*PMSInterface*

All the components after the port in the url are case-sensitive. They should be used in the same case as shown above.

Implementation Change

Applicable to only those OPERA and OXI versions prior to 3.0.1 used the PL/SQL Gateway component for
Apache provided by Oracle to facilitate the http communication. This mandated the clients to use FormEncode data streams while sending data to OXI's communication http server. Also it had issues handling https protocol and there are message size limitations. Though this is still supported, it has been deprecated and being replaced by the servlet based communication module which does not accept form-encoded data, but text/xml plain text data. At the time of this writing, PL/SQL Gateway based http communication is supported till version 3.6 of OPERA and OXI. It may not be supported in future versions. Servlet based communication module is supposed to resolve the following critical shortcomings: Ability to support https and the ability to support messages larger than 32000 bytes.

URL Components

All italicized underlined words in the URL given above must be replaced with actual values. Description of those fields follow:

OperaHTTPServer	Replace with the IP address or host name of the OPERA HTTP server designated for communication
Port	HTTP service's listening port on the OPERA HTTP server. Optional. Specify if different from standard port 80.

URL Parameters

Notes : Parameter names and values are case sensitive. Parameter names must be in camel case as shown below and values must be in upper case.

Name	***Description***	
interfaceName	Interface id assigned to the external system.	
propertyName	Property code used by the external system to identify the property. This should be the same as the value configured in external resort field of OXI's interface setup.	
messageType	Type of the XML message. Refer to OXI as Client: Receive Data from External System section for valid values.	
transactionId	For RESULT messages:	OXI transaction id of the original message received from OXI.
	For other messages:	External system's unique transaction identifier. Can be alphanumeric up to 100 characters long. To make it user and support friendly, it is recommended to use numeric values up to 9 digits long.
status	NEW :	New non-result message.
	SUCCESS :	Only for result messages. To indicate successful processing of original message received from OXIHUB by the external system.

	FAILED :	Only for result messages. To indicate unsuccessful processing of original message received from OXIHUB by the external system.

HTTP Response

Result	***Status Code***	***Content-Type***	***Content***
Message accepted	200 OK	text/plain	Brief status message.
Invalid request from client	400 Bad Request	text/plain	Error message
Error reported by OXI while processing request	500 Internal Server Error	text/plain	Error message

Sample URL

http://oxihub.ft.micros.com:5868/Operajserv/OXIServlets/PMSInterface?interfaceName=ORS&propertyNam
e=MEXICO&messageType=PROFILE&transactionId=60528&status=NEW

Sample POST Data

Content-type : text/xml

Content

```
<?xml version = '1.0'?>
<?Label MEXICO|PROFILE|60528|NEW?>
<Profile profileType="CORPORATE" gender="UNKNOWN"
xmlns="profile.fidelio.2.0">
   <profileID>1641585</profileID>
   <creatorCode>SUPERVISOR</creatorCode>
   <createdDate>2004-06-09T15:29:48.000</createdDate>
<lastUpdaterCode>SUPERVISOR</lastUpdaterCode>
   <lastUpdated>2004-06-09T15:30:22.000</lastUpdated>
   <genericName>Master Account</genericName>
   <IndividualName>
      <nameSur>Master Account</nameSur>
   </IndividualName>
   <PostalAddresses>
      <PostalAddress addressType="BUSINESS">
         <address1>666 Master Account from Profile</address1>
         <city>Naples</city>
         <stateCode>FL</stateCode>
         <postalCode>34101</postalCode>
         <countryCode>US</countryCode>
<mfPrimaryYN>Y</mfPrimaryYN>
```

```
      </PostalAddress>
   </PostalAddresses>
   <PhoneNumbers>
      <PhoneNumber phoneNumberType="BUSINESS">
         <phoneNumber>1237890</phoneNumber>
         <mfPrimaryYN>Y</mfPrimaryYN>
      </PhoneNumber>
   </PhoneNumbers>
   <mfResort>MEXICO</mfResort>
   <mfAllowMail>NO</mfAllowMail>
   <mfAllowEMail>NO</mfAllowEMail>
   <mfGuestPriv>NO</mfGuestPriv>
   <mfAllowPhone>0</mfAllowPhone>
   <mfAllowSMS>0</mfAllowSMS>
</Profile>
```

OXI as Server: Receive Data from OXI

Since OXI can interface with multiple interfaces for multiple properties, external systems must identify themselves and specify the property for which they are querying in the URL parameters.

HTTP Operation : GET
Content-type : text/xml
Character-encoding : UTF-8
URL : http://*OperaHTTPServer*:*Port*/*Operajserv*/*OXIServlets*/*PMSInterface*

URL Components

All italicized underlined words in the URL given above must be replaced with actual values. Description of those fields follow:

OperaHTTPServer	Replace with the IP address or host name of the OPERA HTTP server designated for communication
Port	HTTP service's listening port on the OPERA HTTP server. Optional. Specify if different from standard port 80.

URL Parameters

Notes : Parameter names and values are case sensitive. Parameter names must be in camel case as shown below and values must be in upper case.

Name	***Description***
interfaceName	Interface id assigned to the external system.
propertyName	Property code used by the external system to identify the property. This should be the same as the value configured in external resort field of OXI's interface setup.

Sample URL

```
URL:
http://nploxidb/Operajserv/OXIServlets/PMSInterface?interfaceName=
ORS&propertyName=SLB Request Method : GET
```

HTTP Response - Message is available for property

HTTP Return Code	200 OK
Response Stream	XML message
Response Header Fields	
Content-Type	text/xml
Content-Length	Not present
Title	*propertyName*|*messageType*|*transactionId*|*status* These values are similar to the parameters passed to OXIHUB while sending data to OXIHUB.

Sample Response

```
Response code    : 200
Response message : OK
```

Header Fields

```
1. Connection: [close]
2. title: [SLB|PROFILE|73|NEW]
3. cache-control: [no-control]
4. Date: [Fri, 21 May 2004 17:52:23 GMT]
5. Pragma: [no-cache]
6. Server: [Oracle HTTP Server Powered by Apache/1.3.19 (Win32)
   mod_ssl/2.8.1
OpenSSL/0.9.5a mod_oprocmgr/1.0 mod_perl/1.25]
7. Content-Type: [text/xml; charset=UTF-8]
8. null: [HTTP/1.1 200 OK]
9. Transfer-Encoding: [chunked]
```

Output Data Stream

```
<?xml version = '1.0'?>
<?Label SLB|PROFILE|73|NEW?>
<Profile profileType="GUEST" gender="UNKNOWN"
xmlns="profile.fidelio.1.2">
   <profileID>1505084539</profileID>
   <creatorCode>OXI-OPERA</creatorCode>
   <createdDate>2004-05-18T18:00:27.000</createdDate>
<lastUpdaterCode>OXI-OPERA</lastUpdaterCode>
   <lastUpdated>2004-05-18T18:00:29.000</lastUpdated>
   <genericName>Noshow</genericName>
   <IndividualName>
      <nameFirst>First</nameFirst>
      <nameSur>Noshow</nameSur>
   </IndividualName>
   <primaryLanguageID>E</primaryLanguageID>
```

```
   <PostalAddresses>
      <PostalAddress addressType="HOME">
         <countryCode>MX</countryCode>
<mfPrimaryYN>Y</mfPrimaryYN>
      </PostalAddress>
   </PostalAddresses>
   <mfResort>SLB</mfResort>
   <mfResortProfileID>34676</mfResortProfileID>
   <mfNameCode>99565</mfNameCode>
   <mfAllowMail>NO</mfAllowMail>
   <mfAllowEMail>NO</mfAllowEMail>
   <mfGuestPriv>NO</mfGuestPriv>
</Profile>
```

HTTP Response - Message is not available for property

HTTP Return Code	200 OK
Response Stream	Not applicable
Response Header Fields	
Content-Type	Not Applicable
Content-Length	0
Title	Not present
PS	In future, System may return [204][No Content] to indicate that there is no data available.

Sample Response

```
Response code    : 200
Response message : OK
```

Header Fields

```
1. Content-Length: [0]
2. Connection: [close]
3. cache-control: [no-control]
4. Date: [Fri, 21 May 2004 17:51:30 GMT]
5. null: [HTTP/1.1 200 OK]
6. Pragma: [no-cache]
7. Server: [Oracle HTTP Server Powered by Apache/1.3.19 (Win32)
   mod_ssl/2.8.1
OpenSSL/0.9.5a mod_oprocmgr/1.0 mod_perl/1.25]
8. Content-Type: [text/plain]
```

HTTP Response - Error occurred while processing the GET request

HTTP Return Code	400 Bad Request 500 Internal Server Error
Response Stream	Applicable error message

Response Header Fields	
Content-Type	text/plain
Content-Length	Not Applicable
Title	Not Applicable

Sample Response

```
Response code    : 400
Response message : Bad Request
```

Header Fields

```
1. Connection: [close]
2. cache-control: [no-control]
3. Date: [Fri, 21 May 2004 18:27:39 GMT]
4. Pragma: [no-cache]
5. Server: [Oracle HTTP Server Powered by Apache/1.3.19 (Win32)
   mod_ssl/2.8.1
OpenSSL/0.9.5a mod_oprocmgr/1.0 mod_perl/1.25]
6. Content-Type: [text/plain]
7. null: [HTTP/1.1 400 Bad Request]
8. Transfer-Encoding: [chunked]
```

Error Stream

```
Dequeue error for [SLB]: java.sql.SQLException: ORA-04068:
existing state of packages has been discarded
ORA-04061: existing state of package body
"OXIHUB46_D.INT_COMM_MAIN" has been invalidated ORA-04065: not
executed, altered or dropped package body
"OXIHUB46_D.INT_COMM_MAIN"
ORA-06508: PL/SQL: could not find program unit being called
ORA-06512: at
"OXIHUB46_D.XMLDEQUEUECLOB2", line 22
ORA-06512: at line 1
```

FTP

In order to utilize data transfer via FTP protocol, a FTP server must exist that can receive PUT and GET commands from another system.

- OXI transfers data from the external system into the Inbound queue from where the OXI download processor takes over.
- OXI produces upload messages from the OPERA business events and stores these in the Outbound queue using its upload processor and an FTP transfer mechanism to send the messages to the external system.

External system has an FTP server

This is the standard handling.

- The only way we are transmitting files between OXI and an external system using FTP
- OXI does not need any additional hardware as it is functioning as the FTP client application.
- The external system has a FTP server installed
- The external FTP server owner provides the OXI user with the hostname, username, password and folder information for put and get
- OXI sends its data load to the external FTP server using the PUT command. External system is responsible for processing messages delivered by OXI.
- OXI requests data from external system using the GET command. This is done in frequent, definable intervals. External system is responsible for placing available messages in the designated folder on the FTP server for OXI to download.
- The advantage is that the external vendor can potentially reuse communication parts that they have already in place.

Messages into OPERA

- Reservation is created in external system.
- OXI requests data from external system FTP server using FTP/GET.
- OXI receives the entire XML message, if available, and parses it, performs necessary conversion and writes the data to the OPERA database.
- The reservation in the OPERA database is accessible from any OPERA workstation.

Messages from OPERA

- Depending on the business event configuration, the OPERA user creates an activity in OPERA, which is captured in form of a business event.
- The OXI upload process dequeues the business event, validates the record, and applies data conversion.
- The result is stored as XML file.
- OXI sends the XML file to the external FTP server using FTP/PUT.

File System

In order to utilize the file transfer communication method, the external system vendor needs to establish communication to the OXI interface PC and place data into a defined directory on that PC or the surrounding network.

Messages into OPERA

- A message is created in the external system and sent to the OPERA property or data center to be stored in a defined import directory.
- The OXA C/C++ program running on the Windows interface PC reads the XML file from the directory, does pre-validation of data elements, and writes the data into temporary XML tables.
- The OXA C/C++ program calls OXI PL/SQL code to read the data from the temporary XML tables.
- The normal OXI download processor handling begins.

Messages from OPERA

- An activity happens in OPERA and business events are created.
- The OXI upload processor processes the data and stores it in temporary XML tables.
- The OXA C/C++ program places the file into a specified export directory.
- The external system process picks up the XML message and establishes connection to the external system to send the message.