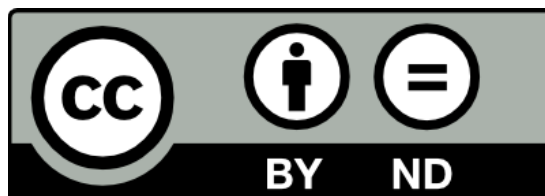




AlpineBits is a specification of an OTA-based interface that is specially tailored for alpine tourism. Its purpose is to facilitate data exchange in the tourism sector.

compatible with version 2010A of the OpenTravel Schema by the OpenTravel Alliance.

www.alpinebits.org



© AlpineBits Alliance. This document is licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License [u](https://creativecommons.org/licenses/by-nc/3.0/).

Permissions beyond the scope of this license may be available at www.alpinebits.org

About the AlpineBits Alliance

The “AlpineBits Alliance” is a group of SME operating in the touristic sector working together to innovate and open the data exchange in the alpine tourism, and therefore to optimize the online presence, sales and marketing efforts of the hotels and other accommodations in the alpine territory and also worldwide.

AlpineBits Alliance

Via Bolzano 63/A
39057 Frangarto / Appiano s.s.d.v. (BZ) - ITALY
VAT Reg No: IT02797280217
www.alpinebits.org
info@alpinebits.org

AlpineBits Contributors

This specification is the result of an open discussion and development process by several different entities:

Altea Software Srl - www.altea.it
Brandnamic GmbH - www.brandnamic.com
Internet Consulting GmbH - www.inetcons.it
LOGon KG - www.logon.it
Marketing Factory GmbH - www.marketingfactory.it
Nexteam Srl - Booking Expert - www.bookingexpert.it
PCS Hotelsoftware GmbH - www.pcs-phoenix.com
Peer GmbH - www.peer.biz
Seekda GmbH - www.seekda.com
SiMedia GmbH - www.simedia.com
TIS innovation park - www.tis.bz.it
and all the members of the online discussion group: <https://groups.google.com/forum/#!forum/alpinebits>

Author of this document: Chris Mair - www.1006.org

Disclaimer

This specification is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY.

If you find errors or you have proposals for enhancements, do not hesitate to contact us using the online discussion group: <https://groups.google.com/forum/#!forum/alpinebits>.

Document Change Log

Important note: make sure to have the latest version of this document! The latest version is available from <http://www.alpinebits.org>.

protocol version	doc. release date	description
2013-04	2013-05-24	<p>Status:</p> <ul style="list-style-type: none"> official release <p>Updates:</p> <ul style="list-style-type: none"> Section 2 (HTTPS layer): added information regarding the new X-AlpineBits-ClientID and X-AlpineBits-ClientProtocolVersion fields in the HTTP header Section 3.2 (capabilities): added capability for FreeRoom deltas Section 4 (Intro): changed the text a bit to make it clearer that AlpineBits does indeed support booking requests and not only requests for quotes Section 4.1 (FreeRooms): added the possibility to send partial information (deltas); added warning response; much improved description of the response in general Section 4.2 (GuestRequests): slightly improved the description of the response in case of error Section 4.3 (Simple Packages): added limitation (just one Hotel per request); added warning response; much improved description of the response in general; clarified text to explicitly state that it is not allowed to mix package add and delete requests in a single message Appendix A (example code): this document should be language neutral so the code that used to be here has been removed with a message to check the official AlpineBis site (with the current release the code is still in the documentation kit, however) Appendix B (compatibility matrix): new appendix
2012-05b	2012-10-01	<p>Status:</p> <ul style="list-style-type: none"> official release <p>Updates:</p> <ul style="list-style-type: none"> OTA compatibility: the attribute Thu is renamed to Thur and the attribute wed is renamed to weds SimplePackages: the element Image is listed as mandatory in the table as it already was in the text and schema files SimplePackages: The element RateDescription is listed as non-repeatable in the table as it already was in the text and schema files
2012-05	2012-05-31	<p>Status:</p> <ul style="list-style-type: none"> official release <p>Updates:</p> <ul style="list-style-type: none"> major rewrite of the text FreeRooms: action OTA_HotelAvailNotif is no longer mandatory <p>Additions:</p> <ul style="list-style-type: none"> GuestRequests: reservation inquiries SimplePackages: package availability notifications <p>Deletions:</p> <ul style="list-style-type: none"> None
2011-11	2011-11-18	Minor alterations and release under Creative Commons Attribution-NoDerivs 3.0 Unported License
2011-10	2011-10-20	production release with minor alterations
2011-09	2011-09-08	first draft of redesigned version (using POST instead of SOAP)
2010-10	2010-10-20	second draft
2010-08	2010-08-01	first draft

Table of Contents

- [1. Introduction](#)
- [2. The HTTPS request and response structure](#)
 - [Implementation tips and best practice](#)
- [3. Housekeeping actions](#)
 - [3.1. Query the server version](#)
 - [3.2. Query the server capabilities](#)
 - [3.3. Unknown or missing actions](#)
 - [3.4. Implementation tips and best practice](#)
- [4. Data exchange actions](#)
 - [4.1. FreeRooms: Room availability notifications](#)
 - [Client Request](#)
 - [Sending full information vs. sending partial information](#)
 - [Client Request Elements and Attributes](#)
 - [Server response](#)
 - [Success](#)
 - [Warning](#)
 - [Error](#)
 - [Server Response Elements and Attributes](#)
 - [Implementation tips and best practice](#)
 - [4.2. GuestRequests: requests for a quote and booking requests](#)
 - [Client Request](#)
 - [Client Request Elements and Attributes](#)
 - [Server Response](#)
 - [Error](#)
 - [Success](#)
 - [Server Response Elements and Attributes](#)
 - [Implementation tips and best practice](#)
 - [4.3. SimplePackages: Package availability notifications](#)
 - [Client Request \(notify package availability\)](#)
 - [Client Request \(notify that a package is no longer available\)](#)
 - [Client Request Elements and Attributes](#)
 - [Server response](#)
 - [Success](#)
 - [Warning](#)
 - [Error](#)
 - [Server Response Elements and Attributes](#)
 - [Implementation tips and best practice](#)
- [A. Example code snippets](#)
- [B. Protocol Version Compatibility](#)
 - [2012-05b client talking to 2013-04 server](#)
 - [Housekeeping](#)
 - [FreeRooms](#)
 - [GuestRequests](#)
 - [SimplePackages](#)
 - [2013-04 client talking to 2012-05b server](#)
 - [Housekeeping](#)
 - [FreeRooms](#)
 - [GuestRequests](#)
 - [SimplePackages](#)
- [C. Links](#)

1. Introduction

This documents describes a standard for exchanging traveling and booking information, called **AlpineBits**.

AlpineBits builds upon established standards:

- client-server communication is done through stateless HTTPS (the client POSTs data to the server and gets a response) with basic access authentication [1](#) and
- the traveling and booking information are encoded in XML following version 2010A of the OpenTravel Schema [3.4.5](#) (from here on called OTA2010A) by the OpenTravel Alliance [2](#).

At the current version of the standard, the scope of AlpineBits covers exchanging the following types of information:

- room availability notifications,
- reservation inquiries and
- package availability notifications.

AlpineBits relies on its underlying transport protocol to take care of security issues. Hence **the use of HTTPS is mandatory**.

2. The HTTPS request and response structure

An AlpineBits compliant server exposes a **single** HTTPS URL. Clients send POST requests to that URL.

The POST request **must** transmit the access credentials using **basic access authentication**.

The HTTPS header of the POST request **must** contain an X-AlpineBits-ClientProtocolVersion field. The value of this field is the protocol version supported by the client (see the first column of the changelog table). A server that does not receive the field will simply conclude the client speaks a protocol version preceding the version when this field was introduced (2013-04).

The HTTPS header of the POST request **may** contain an X-AlpineBits-ClientID field. The value of this field is an arbitrary string a server implementer **might** want to use to identify the client software version or installation ID.

The POST request **must** follow the multipart/form-data encoding scheme, as commonly used in the context of HTML forms for file uploads.

The POST requests **must** have **at least** one parameter named **action**. Depending on the value of **action**, one additional parameter named **request** might be required.

Following is a capture of an example POST. In this example, the value of **action** is the string `OTA_HotelAvailNotif`, indicating the client wishes to perform a room availability notification. The value of **request** is an XML document (not fully shown).

```
POST / HTTP/1.1
Authorization: Basic Y2hyaXM6c2VjcmV0
Host: localhost
Accept: */*
X-AlpineBits-ClientProtocolVersion: 2013-04
X-AlpineBits-ClientID: sample-client.php v. 2013-04 1.0
Content-Length: 1989
Expect: 100-continue
Content-Type: multipart/form-data; boundary=-----9d7042ecb251

-----9d7042ecb251
Content-Disposition: form-data; name="action"
```

```
OTA_HotelAvailNotif
-----9d7042ecb251
Content-Disposition: form-data; name="request"

<?xml version="1.0" encoding="UTF-8"?>

<OTA_HotelAvailNotifRQ
  [...]
</OTA_HotelAvailNotifRQ>
-----9d7042ecb251--
```

Note the Authorization field, with the username/password string (`chris:secret`) encoded in base64 (`Y2hyaXM6c2VjcmV0`) as defined by the basic access authentication standard [1](#).

Also note the X-AlpineBits-ClientProtocolVersion and X-AlpineBits-ClientID fields and the multipart content with the values of parameters `action` and `request`.

As a result of the POST request, the server answers with a response. Of course, the content of the response depends on the POSTed parameters, in particular the value of `action`. AlpineBits currently identifies two kinds of actions: the so-called housekeeping actions explained in section 3 and the actual data exchange actions explained in section 4.

The expected status code of the response is 200 (Ok), indicating the server could authenticate the user (with or without being able to actually process any action).

In case of authentication failure (either an invalid or missing username/password or a value of X-AlpineBits-ClientID that is not acceptable to the server) the status code is 401 (Authorization Required) and the content is `ERROR`, followed by a colon (`:`) and an error message indicating the reason of the failure, such as no username/password was provided or the password expired, etc. Regarding the X-AlpineBits-ClientID, a server chooses to require the ID or to ignore the ID. If the server chooses to ignore the ID, it **must** do so silently, i.e. it **must** not return a 401 status because of the presence of the ID.

In case of internal server problem the status code is 500 (Internal Server Error).

An AlpineBits client **must** be able to handle these status codes. It **should** retry a request that has failed (error code 500 or timeout) and only escalate the failure after 2 retries with an appropriate delay.

Implementation tips and best practice

- For maximum compatibility across different implementations AlpineBits implementers are asked to handle POST requests using a supporting API in their language of choice. See appendix A for examples.
- All POSTs to an AlpineBits server are sent to a single URL. However, a server implementer might make more than one URL available for independent servers, such as servers with support for different versions:
 - `https://server.example.com/alpinebits/2011-11`
 - `https://server.example.com/alpinebits/2012-05b`
 - etc...

3. Housekeeping actions

These actions allow the client to query the server version and capabilities. An AlpineBits server **must** be able to handle **both**.

It is the **client's responsibility** to ensure the server can handle the actions it intends to perform. Clients are therefore invited to query the server's capabilities before performing the data exchange actions described in section 4.

The following table lists all available housekeeping actions.

usage	mandatory	available since version	parameter action (string)	parameter request (string)	server response (string)
a client queries the server version	YES	2011-11	getVersion	(not sent)	the server version
a client queries the server capabilities	YES	2011-11	getCapabilities	(not sent)	the server capabilities

3.1. Query the server version

A clients performs this action to query the server version. The values of **action** is the string `getVersion`. The parameter **request** is not specified.

The response content is the string `OK:` followed by the protocol version supported by the server (see the first column of the changelog table, e.g. 2011-11 for the first release version of AlpineBits).

3.2. Query the server capabilities

A clients performs this action to query the server capabilities. The values of **action** is the string `getCapabilities`. The parameter **request** is not specified.

The response is a string starting with `OK:` followed by a list of comma separated tokens each of which indicates a single capability of the server.

AlpineBits specifies the following capabilities:

- `action_getVersion`
the server implements the `getVersion` action
- `action_getCapabilities`
the server implements the `getCapabilities` action
- `action_OTA_HotelAvailNotif`
the server implements handling room availability notifications
- `OTA_HotelAvailNotif_accept_rooms`
for room availability notifications, the server accept booking limits for specific rooms
- `OTA_HotelAvailNotif_accept_categories`
for room availability notifications, the server accept booking limits for categories of rooms
- `OTA_HotelAvailNotif_accept_los`
for room availability notifications, accept length of stay restrictions
- `OTA_HotelAvailNotif_accept_dow`
for room availability notifications, accept day of the week restrictions
- `OTA_HotelAvailNotif_accept_deltas`
for room availability notifications, accept partial information (deltas)

- `action_OTA_Read`
the server implements handling reservation inquiries
- `action_OTA_HotelRatePlanNotif`
the server implements handling package availability notifications

AlpineBits **requires** a server to support at least all mandatory housekeeping actions.

All other capabilities are optional. It is a **client's responsibility** to check for server capabilities before trying to use them. A server implementation is free to ignore information that requires a capability it doesn't declare. A server **must**, however, implement all capabilities it declares.

3.3. Unknown or missing actions

Upon receiving a request with an unknown or missing value for action, the server response is the string:
`ERROR:unknown or missing action.`

3.4. Implementation tips and best practice

- Since the `getCapabilities` request is authenticated it's possible for a server to announce different capabilities to different users.
- In the case a server is able to deal both with categories and rooms data, but only accepts restrictions such as DOW and LOS for one of the two, the creation of two independent servers (URLs) is possible.
- OTA requires the root element of an XML document to have a version attribute. As regards AlpineBits, the value of this attribute is irrelevant.

4. Data exchange actions

These actions allow the actual exchange of data between client and server.

The parameter **request** is mandatory. Both, the client request and the server response are XML documents following OTA2010A as specified in the following table.

known as	usage	mandatory	since version	parameter action (string)	parameter request (XML document)	server response (XML document)
FreeRooms	a client sends room availability notifications to a server	NO	2011-11	OTA_HotelAvailNotif	OTA_HotelAvailNotifRQ	OTA_HotelAvailNotifRS
GuestRequests	a client sends a request to receive requests for a quote or booking requests from the server	NO	2012-05	OTA_Read:GuestRequests	OTA_ReadRQ	OTA_ResRetrieveRS
SimplePackages	a client sends package availability notifications to a server	NO	2012-05	OTA_HotelRatePlanNotif:SimplePackages	OTA_HotelRatePlanNotifRQ	OTA_HotelRatePlanNotifRS

AlpineBits **requires** all XML documents to be encoded in **UTF-8**.

The business logic of an AlpineBits server, i.e. how the server processes and stores the information it receives is implementation-specific.

The format of the requests and responses is, however, exactly specified.

First of all the requests and responses **must** validate against the OTA2010A schema.

Since OTA is very flexible regarding mandatory / optional elements, AlpineBits adds extra requirements about exactly which elements and attributes are required in a request.

If these are not present, a server's business logic is bound to fail and will return a response indicating error even though the request is valid OTA2010A.

OTA2010A, for instance allows OTA_HotelAvailNotifRQ requests that do not indicate the hotel, this might make perfect sense in some context, but an AlpineBits server will return an error if that information is missing from the request.

To aid developers, an AlpineBits XML Schema file and a set of Relax NG files are provided as integral part of the specification in the AlpineBits documentation kit.

The Relax NG file set is stricter than the XML schema file. First, there is a RelaxNG file for each request and response type, so the nodes can be validated in a more specific way. Second, RelaxNG is intrinsically more powerful in expressing constraints that express how elements and attributes depend on each other.

Both, the XML Schema file and the Relax NG files, are stricter than OTA2010A in the sense that all documents that validate against AlpineBits will also validate against OTA2010A, but not vice versa.

The AlpineBits documentation kit also provides a sample file for each of the request and response documents.

The latest AlpineBits documentation kit for each protocol version is available from the official AlpineBits website.

4.1. FreeRooms: Room availability notifications

When the value of the **action** parameter is `OTA_HotelAvailNotif` the client intends to send room availability notifications to the server.

A server that supports this action **must** support at least one out of the two capabilities `OTA_HotelAvailNotif_accept_rooms` or `OTA_HotelAvailNotif_accept_categories`. This way the server indicates whether it can handle the availability of rooms at the level of distinct rooms, at the level of categories of rooms or both.

Client Request

The parameter **request** must contain an `OTA_HotelAvailNotifRQ` document.

Following is a detailed description of the form of this document using one of the sample files from the documentation kit.

First, consider the outer part of the document:

```
<?xml version="1.0" encoding="UTF-8"?>

<OTA_HotelAvailNotifRQ
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelAvailNotifRQ.xsd"
  Version="1.002">

  <AvailStatusMessages HotelCode="123" HotelName="Frangart Inn">

    <!-- ... see below ... -->

  </AvailStatusMessages>

</OTA_HotelAvailNotifRQ>
```

sample-FreeRooms-OTA_HotelAvailNotifRQ.xml - outer part

An `OTA_HotelAvailNotifRQ` may contain just **one** `AvailStatusMessages` (note the plural) element, hence at most **one** hotel can be dealt with in a single request.

AlpineBits requires the attributes `HotelCode` or `HotelName` to be present (and match information in the server's database). The fictitious hotel in the example is the "Frangart Inn" with code "123". Specifying both — code and name — is redundant, but allowed, as long as both are consistent.

AlpineBits **requires** a match of `HotelCode`, `HotelName` to be **case sensitive**.

Second, consider the inner part of the example that contains `AvailStatusMessage` (note the singular) elements for three different rooms.

Let's start with the availabilities for room 101S.

```
<AvailStatusMessage BookingLimit="1" BookingLimitMessageType="SetLimit">

  <StatusApplicationControl Start="2010-08-01" End="2010-08-10"
    InvCode="101S" IsRoom="true"/>

</AvailStatusMessage>

<AvailStatusMessage BookingLimit="1" BookingLimitMessageType="SetLimit">

  <StatusApplicationControl Start="2010-08-21" End="2010-08-30"
```

```
        InvCode="101S" IsRoom="true"/>
</AvailStatusMessage>
```

sample-FreeRooms-OTA_HotelAvailNotifRQ.xml - inner part (1/3)

The **IsRoom** attribute is true, indicating we're dealing with a specific room. Hence **InvCode** indicates the room identifier (101S). Alternatively, a false value for **IsRoom** would indicate a category of rooms with **InvCode** being the category name.

AlpineBits **requires** a match of **InvCode** to be **case sensitive**.

An AlpineBits server **must** be able to treat **at least** one case out of the two cases for **IsRoom** (true or false). A client should perform the `getCapabilities` action to find out whether the server treats the room case (token `OTA_HotelAvailNotif_accept_rooms`), the category case (token `OTA_HotelAvailNotif_accept_categories`) or both.

Mixing rooms and categories in a single request is **not** allowed. An AlpineBits server **must** return an error if it receives such a mixed request.

The attribute **Start** and **End** indicate that room 101S is available from 2010-08-01 to 2010-08-10 and from 2010-08-21 to 2010-08-30.

Regarding the first interval, this means the earliest possible check-in is 2010-08-01 afternoon and latest possible check-out is 2010-08-11 morning (maximum stay is 10 nights).

Since there are no further restrictions, check-ins **after** 2010-08-01 and stays of **less** than 10 nights are allowed as well, provided the check-out is not later than 2010-08-11 morning.

Idem for the other block of 10 nights from 2010-08-21 to 2010-08-30 (latest check-out is 2010-08-31 morning).

Since a specific room is indicated here, the only meaningful value of **BookingLimit** is 0 or 1 (the same room can not be available more than once). In the category case, numbers larger than 1 would also be allowed.

BookingLimit numbers are always interpreted to be absolute numbers. Differential updates are not allowed.

Note that AlpineBits does **not allow** **AvailStatusMessage** elements with overlapping periods. This implies that the order of the **AvailStatusMessage** elements doesn't matter. It is a **client's responsibility** to avoid overlapping. An AlpineBits server's business logic **may** identify overlapping and return an error or **may** proceed in a implementation-specific way.

AlpineBits **requires** the **AvailStatusMessage** attributes **BookingLimit** and **BookingLimitMessageType** to be given. It also requires exactly one **StatusApplicationControl** element with attributes **Start**, **End**, **InvCode** and **IsRoom** for each **AvailStatusMessage** element. It **must** return an error if any of these are missing.

Next: the availabilities for room 102.

```
<AvailStatusMessage BookingLimit="1" BookingLimitMessageType="SetLimit">
  <StatusApplicationControl Start="2010-08-14" End="2010-08-27"
    InvCode="102" IsRoom="true" Sat="true" />
  <LengthsOfStay>
    <LengthOfStay MinMaxMessageType="SetMinLOS" Time="7" TimeUnit="Day"/>
    <LengthOfStay MinMaxMessageType="SetMaxLOS" Time="7" TimeUnit="Day"/>
  </LengthsOfStay>
</AvailStatusMessage>
```

```
<LengthsOfStay>
<AvailStatusMessage>
```

sample-FreeRooms-OTA_HotelAvailNotifRQ.xml - inner part (2/3)

Room 102 is available from 2010-08-14 to 2010-08-27, but allows arrivals only on Saturdays and requires a stay of 7 nights.

That means there are two blocks that can be sold:

- arrival Saturday 2010-08-14, stay 7 nights, departure Saturday 2010-08-21 morning
- arrival Saturday 2010-08-21, stay 7 nights, departure Saturday 2010-08-28 morning

Restrictions on arrivals to certain days of week (DOWs) must be indicated by setting the corresponding DOW attributes to true: **Mon, Tue, Weds, Thur, Fri, Sat** or **Sun** (note that two of these attributes have 4 letters).

A DOW attribute set to false has the same meaning as if missing. If neither of these attributes is set to true, there are no DOW restrictions at all.

An AlpineBits server **might** ignore DOW restrictions silently. It is the **client's responsibility** to check whether the server supports DOW restrictions performing the `getCapabilities` action: the token to look for is `OTA_HotelAvailNotif_accept_dow`.

Restrictions on length of stay (LOS) must be indicated using one or two **LengthOfStay** elements with the **MinMaxMessageType** attribute set to `SetMinLOS` or `SetMaxLOS`. AlpineBits expects the **TimeUnit** to always be `Day`.

Note that LOS restrictions are considered to be "on arrival" not "stay through". That means only the LOS restriction of the arrival day applies.

Again, an AlpineBits server **might** ignore LOS restrictions silently. It is the **client's responsibility** to check whether the server supports LOS restrictions performing the `getCapabilities` action: the token to look for is `"OTA_HotelAvailNotif_accept_los"`.

The last **AvailStatusMessage** in the example is about room 103.

```
<AvailStatusMessage BookingLimit="1" BookingLimitMessageType="SetLimit">
  <StatusApplicationControl Start="2010-08-01" End="2010-08-31"
    InvCode="103" IsRoom="true"/>
</AvailStatusMessage>
```

sample-FreeRooms-OTA_HotelAvailNotifRQ.xml - inner part (3/3)

Room 103 is available in August (earliest check-in 2010-08-01 afternoon and latest check-out 2010-09-01 morning) with no further restrictions.

Sending full information vs. sending partial information

Note that within the scope of AlpineBits, a server **must** have knowledge about each hotel's complete set of room identifiers and/or complete set of category names. This information is supposed to have been shared in advance! The `OTA_HotelAvailNotifRQ` request document sent by the client **must not** be used to deduce this information.

If a client sends a request as outlined above, AlpineBits **requires** it to mention **every room** (or **every category**) in the request and cover the whole range of dates the client wishes to transmit. In this case the server **must** answer with an error to any request that does not mention all rooms (or categories) or

that contains unknown rooms (or categories). This is necessary in order to safeguard against situations where a server's database is not kept up to date with regard to a hotel's capacity.

However, a client may also send partial information: so called "deltas", if the server supports it.

For deltas, the client **may** send only a subset of rooms (or categories) and/or only information regarding the period where availability has changed since the last request. In any case, the **absolute number** of rooms (or categories) available **must** be sent: 0 or 1 for rooms and 0 .. N for categories.

A client **must** indicate the fact that it is sending delta information by setting the **MessageContentCode** attribute of the root element (**OTA_HotelAvailNotifRQ**) to the fixed value 5 ("Segment/room availability" according to OTA).

The outer part of a delta request thus becomes:

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_HotelAvailNotifRQ
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelAvailNotifRQ.xsd"
  Version="1.002"
  MessageContentCode="5">

  <AvailStatusMessages HotelCode="123" HotelName="Frangart Inn">

    <!-- [...] -->

  </AvailStatusMessages>
</OTA_HotelAvailNotifRQ>
```

sample-FreeRooms-OTA_HotelAvailNotifRQ-delta.xml - outer part

AlpineBits **recommends** that Implementers that use delta requests **should** send the full set of information periodically.

A server that supports delta requests **must** indicate so via the `OTA_HotelAvailNotif_accept_deltas` capability. As always, it is the **client's responsibility** to check whether the server supports deltas before trying to send them.

Client Request Elements and Attributes

Name of Element / <i>Attribute</i>	Annotations	Mand	Rept	Children of/Attribute of
OTA_HotelAvailNotifRQ		yes	no	- (root element)
MessageContentCode	set to "5" if and only if the request contains partial information (deltas)	*	no	OTA_HotelAvailNotifRQ
AvailStatusMessages		yes	no	OTA_HotelAvailNotifRQ
HotelCode	either this or HotelName are mandatory	*	no	AvailStatusMessages
HotelName	either this or HotelCode are mandatory	*	no	AvailStatusMessages
AvailStatusMessage		yes	yes	AvailStatusMessages
BookingLimit		yes	no	AvailStatusMessage
BookingLimitMessageType	mandatory value: "SetLimit"	yes	no	AvailStatusMessage
StatusApplicationControl		yes	no	AvailStatusMessage
Start	First day available for checkin/stay of guest.	yes	no	StatusApplicationControl
End	Last day available for checkin/stay of guest. Departure is the morning after this date.	yes	no	StatusApplicationControl
InvCode		yes	no	StatusApplicationControl
IsRoom	"true" for rooms, "false" for categories	yes	no	StatusApplicationControl
Mon	"true" or "false"	no	no	StatusApplicationControl
Tue	"true" or "false"	no	no	StatusApplicationControl
Weds	"true" or "false"	no	no	StatusApplicationControl
Thur	"true" or "false"	no	no	StatusApplicationControl
Fri	"true" or "false"	no	no	StatusApplicationControl
Sat	"true" or "false"	no	no	StatusApplicationControl
Sun	"true" or "false"	no	no	StatusApplicationControl
LengthsOfStay		no	no	AvailStatusMessage
LengthOfStay		no	yes	LengthsOfStay
MinMaxMessageType	"SetMinLOS" or "SetMaxLOS"	yes	no	LengthOfStay
Time		yes	no	LengthOfStay
TimeUnit	mandatory value: "Day"	yes	no	LengthOfStay

Server response

The server will send a response indicating the outcome of the request. The response is a OTA_HotelAvailNotifRS document. Three types of outcome are possible: success, warning or error.

Success

The request was accepted and processed successfully. The client does **not** need to take any further action.

In this case the OTA_HotelAvailNotifRS response contains nothing but a **single**, empty **Success** element:

```
<?xml version="1.0" encoding="UTF-8"?>

<OTA_HotelAvailNotifRS
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelAvailNotifRS.xsd"
  Version="1.001">

  <Success/>

</OTA_HotelAvailNotifRS>
```

sample-FreeRooms-OTA_HotelAvailNotifRS-success.xml

Warning

The request was accepted and processed successfully. However, one or more non-fatal problems were detected and added to the server response. The client does **not** need to resend the request, but **must** notify the user or the client implementer regarding the warnings received.

In this case the OTA_HotelAvailNotifRS response contains an empty **Success** element followed by **one or more Warning** elements with the attribute **Type** set to the fixed value 11 ("Advisory" according to OTA) containing a human readable text:

```
<?xml version="1.0" encoding="UTF-8"?>

<OTA_HotelAvailNotifRS
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelAvailNotifRS.xsd"
  Version="1.001">

  <Success/>
  <Warnings>
    <Warning Type="11">
      Warning: last full data set received more than 48 hours ago.
    </Warning>
  </Warnings>

</OTA_HotelAvailNotifRS>
```

sample-FreeRooms-OTA_HotelAvailNotifRS-warning.xml

Error

The request **could not** be accepted or processed successfully. The client **must** take action: it **may** try to resend the message if it has reason to assume the error is transient and occurred for the first time, otherwise it **must** escalate the problem to the user or client implementer.

In this case the OTA_HotelAvailNotifRS response contains **one or more Error** elements each with a **Type** attribute containing a human readable text.

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_HotelAvailNotifRS
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelAvailNotifRS.xsd"
  Version="1.001">
  <Errors>
    <Error Type="404">
      Invalid start/end date combination
    </Error>
  </Errors>
</OTA_HotelAvailNotifRS>
```

sample-FreeRooms-OTA_HotelAvailNotifRS-error.xml

Exactly what errors a server will trap is implementation-specific, but it goes without saying that a server **must** return an error if it cannot process any part of a message.

The value of the type attribute **must** follow the OTA2010A error codes table that comes with the OTA2010A documentation package. The package can be downloaded from the OTA web site [3](#). The file OpenTravel_CodeList_20100322.xls contains the error codes table.

Implementors are **required** to follow that table. Here is a non-exhaustive list of error types that might be encountered when processing FreeRoom requests, taken from the OTA table:

Type	Text
61	Invalid hotel
392	Invalid hotel code
396	Invalid name
375	Hotel not active
135	End date is invalid
136	Start date is invalid
404	Invalid start/end date combination
131	Room/unit type invalid
69	Minimum stay criteria not fulfilled
70	Maximum stay criteria not fulfilled
362	Invalid number of nights

Server Response Elements and Attributes

Name of Element / <i>Attribute</i>	Annotations	Mand	Rept	Children of/Attribute of
OTA_HotelAvailNotifRS		yes	no	- (root element)
Success		no	no	OTA_HotelAvailNotifRS
Warnings		no	no	OTA_HotelAvailNotifRS
Warning		no	yes	Warnings
<i>Type</i>	value = "11"	yes	no	Warning
Errors		no	no	OTA_HotelAvailNotifRS
Error		no	yes	Errors
<i>Type</i>	OTA error codes	yes	no	Error

Implementation tips and best practice

- Note that in the 2011-11 version of AlpineBits the support of this action was mandatory for the server. This is no longer the case.
- Note that sending partial information (deltas) was added with AlpineBits 2013-04.
- An element **StatusApplicationControl** containing all the DOW attributes (**Mon, Tue, Weds, Thur, Fri, Sat, Sun**) set to true has the same meaning as no DOW restriction at all.
- For non-delta requests, since no time frame is explicitly transmitted by the client, a server is encouraged to delete and insert all the information stored in its backend, rather than updating it.
- Please note that the **End** date of an interval identifies the last day and night of the stay. Departure is the morning of the date after the **End** date.
- Please note that the **Type** attribute of the **Warning** element is fixed at 11, while the **Type** attribute of the **Error** element is variable. These two attributes have a different meaning in OTA and should not be confused.

4.2. GuestRequests: requests for a quote and booking requests

When the value of the **action** parameter is `OTA_Read:GuestRequests`, the client (typically the software used by a hotel) sends a request to obtain information from the server regarding **requests for a quote or booking requests**. The typical use case is a portal that has collected such requests from a customer and stores them until the client retrieves them.

Please note that GuestRequests at this point just supports the flow of data from the server to the client:

- GuestRequests **do not aim** at covering the whole customer-hotel interaction upon receiving a request: the hotel is supposed to contact the customer directly to provide a quote or confirm/refuse a booking, as well as to deal with possible cancellations etc.

Client Request

The parameter **request** must contains a `OTA_ReadRQ` document.

For the **mandatory** attributes **HotelCode** and **HotelName** the rules are the same as for room availability notifications (section 4.1).

The element **SelectionCriteria** with the **Start** attribute is **optional**. When given, the server will send only inquiries generated after the **Start** timestamp. Otherwise it will send all inquiries it has on record.

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_ReadRQ xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_ReadRQ.xsd"
  Version="1.001">

  <ReadRequests>
    <HotelReadRequest HotelCode="123" HotelName="Frangart Inn">
      <SelectionCriteria Start="2012-03-21T15:00:00+01:00"></SelectionCriteria>
    </HotelReadRequest>
  </ReadRequests>

</OTA_ReadRQ>
```

sample-GuestRequests-OTA_ReadRQ.xml

Client Request Elements and Attributes

Name of Element / Attribute	Annotations	Mand	Rept	Children of/Attribute of
OTA_ReadRQ		yes	no	- (root element)
ReadRequests		yes	no	OTA_ReadRQ
HotelReadRequest		yes	no	ReadRequests
<i>HotelCode</i>	either this or <i>HotelName</i> are mandatory	*	no	<i>HotelReadRequest</i>
<i>HotelName</i>	either this or <i>HotelCode</i> are mandatory	*	no	<i>HotelReadRequest</i>
SelectionCriteria		no	no	HotelReadRequest

Start		yes	no	SelectionCriteria
-------	--	-----	----	-------------------

Server Response

The server response is a OTA_ResRetrieveRS document indicating the outcome of the request. Two types of outcome are possible: error or success.

Error

If the request **could not** be accepted or processed successfully the OTA_HotelAvailNotifRS response contains one or more **Error** elements each with a **Type** attribute containing a human readable text. The rules for creating error messages are identical to the room availability error case (please see section 4.1).

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_ResRetrieveRS
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_ResRetrieveRS.xsd"
  Version="2.000">
  <Errors>
    <Error Type="392">
      Invalid hotel code
    </Error>
  </Errors>
</OTA_ResRetrieveRS>
```

sample-GuestRequests-OTA_ResRetrieveRS-error.xml

Success

In case of success, the OTA_ResRetrieveRS response will contain a **single**, empty **Success** element followed by a **ReservationsList** element with **zero or more HotelReservation** elements containing the inquiries (zero elements indicate the server has no inquiries for the client at this point).

Shown here is the outer part of a OTA_ResRetrieveRS document.

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_ResRetrieveRS
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_ResRetrieveRS.xsd"
  Version="2.000">
  <Success/>
  <ReservationsList>
    <HotelReservation CreateDateTime="2012-03-21T15:00:00+01:00"
      ResStatus="Book">
      <!-- Type 14 -> Reservation -->
      <UniqueID Type="14" ID="6b34fe24ac2ff810"/>
    </HotelReservation>
  </ReservationsList>
  <RoomStays> <!-- stays, see below --> </RoomStays>
```

```

    <ResGuests>      <!-- customer data, see below -->      </ResGuests>

    <ResGlobalInfo> <!-- additional booking data, see below --> </ResGlobalInfo>

</HotelReservation>

</ReservationsList>

</OTA_ResRetrieveRS>

```

sample-GuestRequests-OTA_ResRetrieveRS-book.xml - outer part

Each **HotelReservation** **must** have the attributes **CreateDateTime** (the timestamp the customer inquiry was collected by the portal). Furthermore the **ResStatus** attribute **must** be set. AlpineBits expects it to be equal to either *Quote* (this is just a customer inquiry for a **quote**) or *Book* (this is a customer **book** request).

The example discussed in this section is a **book** request, the documentation kit also has an example of a **quote** request.

Each **HotelReservation** contains a **mandatory UniqueID** element that the client can use to recognize an inquiry it has already processed. The current release of AlpineBits **requires** the server to not change any information in later exchanges once a unique id has been assigned. This might change in a future versions of AlpineBits, though.

The **UniqueID** element **must** be of **Type 14** (meaning reservation per OTA code table), the attribute is a free text field suitable for uniquely identifying the **HotelReservation**.

The actual data is then split into three parts: each **HotelReservation** contains the elements: **RoomStays**, **ResGuests** and **ResGlobalInfo** (all **mandatory**) discussed in the following paragraphs.

First part: **RoomStays**.

The **RoomStays** element contains **one or more RoomStay** elements, each indicating a desired stay.

```

<RoomStays>

  <RoomStay>

    <RoomTypes>
      <RoomType RoomTypeCode="megasuite"/>
    </RoomTypes>

    <RatePlans>
      <RatePlan RatePlanCode="123456-xyz">
        <!-- Code 1 -> All inclusive -->
        <MealsIncluded MealPlanCodes="1"
          Breakfast="true" Lunch="true" Dinner="true"/>
      </RatePlan>
    </RatePlans>

    <!-- 2 adults + 1 child + 1 child = 4 guests -->
    <GuestCounts>
      <!-- 2 adults -->
      <GuestCount Count="2"/>
      <!-- 1 child -->
      <GuestCount Count="1" Age="9"/>
      <!-- 1 child -->
      <GuestCount Count="1" Age="3"/>
    </GuestCounts>

    <TimeSpan Duration="P11N">
      <StartDateWindow EarliestDate="2012-01-01" LatestDate="2012-01-01"/>
    </TimeSpan>
  </RoomStay>

```

```

    </TimeSpan>

    <Total AmountAfterTax="299" CurrencyCode="EUR"/>

</RoomStay>
</RoomStays>

```

sample-GuestRequests-OTA_ResRetrieveRS-book.xml - RoomStay element

Each **RoomStay** element contains:

- **one RoomType** element (**mandatory** for **book** requests, **optional** for **quote** requests) with a **RoomTypeCode** attribute that defines the desired room category for this stay; in the likely case that the implementation also manages the FreeRooms action, this **RoomTypeCode** attribute should be identical to the **InvCode** attribute used for room categories (see section 4.1)
- **one RatePlan** element with an **RatePlanCode** attribute (**optional**) and **one MealsIncluded** element (**mandatory** for **book** requests, **optional** for **quote** requests): the attributes **Breakfast**, **Lunch** and **Dinner** are set to true or false - all three **must** be given, even where the value is false; the optional attribute **MealPlanCodes** must be set to 1 when indicating an *all inclusive* formula in addition to *full board* (**Breakfast**, **Lunch** and **Dinner** set to true)
- **one GuestCount** element (**mandatory**) indicating the number of all adults (identified by **one GuestCount** element **with no Age** attribute given) and the number of all children (identified by **zero or more GuestCount** element **with Age** attribute given); of course **all** guests must be listed
- **one TimeSpan** element (**mandatory**): see below for explanation
- **one Total** element (**mandatory** for **book** requests, **optional** for **quote** requests) containing the cost after taxes the portal has displayed to the customer, both attributes **AmountAfterTax** and **CurrencyCode** are required.

The **TimeSpan** element deserves a more detailed explanation.

The start date of the desired stay is the date of the check-in. It is **mandatory** and **always** given in windows form using the **StartDateWindow** element with attributes **EarliestDate** and **LatestDate**.

For **quote** requests **LatestDate** might be greater than **EarliestDate** indicating a range of possible start dates.

For **book** requests **LatestDate** must be equal to **EarliestDate** indicating the exact start date.

The **Duration** attribute of the **TimeSpan** is also mandatory and encoded in ISO 8601. AlpineBits accepts only durations given in nights, the form is thus always PxN where x is a number.

If multiple **RoomStay** elements are given, all the **TimeSpan** elements must have exactly the same values.

As a special case, however, **only** for **quote** requests, it is possible to add **at most one optional RoomStay** element that contains **only** the **TimeSpan** element. In this case, this last **TimeSpan** is allowed to have different values (as a matter of fact, they must be different) and it ought to be interpreted by the client as an alternative period with regard to the preceding **RoomStay** element(s).

Second part: ResGuests.

Nested inside the **ResGuests** element is **exactly one Customer** element, giving the data of the primary guest.

The **Gender** attribute is **mandatory**, but can be `Unknown`, besides `Male` or `Female`.

The **BirthDate** attribute (ISO 8601) is **optional**.

The **Language** attribute is **mandatory**, it must follow ISO 639-1 (two-letter lowercase language abbreviation). It identifies the language to be used when contacting the customer.

```
<ResGuests>
  <ResGuest>
    <Profiles>
      <ProfileInfo>
        <Profile>

          <Customer Gender="Male" BirthDate="1980-01-01" Language="de">

            <PersonName>
              <NamePrefix>Herr</NamePrefix>
              <GivenName>Otto</GivenName>
              <Surname>Mustermann</Surname>
              <NameTitle>Dr</NameTitle>
            </PersonName>

            <!-- Code 1 -> Voice -->
            <Telephone PhoneTechType="1" PhoneNumber="+4934567891"/>
            <!-- Code 3 -> Fax -->
            <Telephone PhoneTechType="3" PhoneNumber="+4934567892"/>
            <!-- Code 5 -> Mobile -->
            <Telephone PhoneTechType="5" PhoneNumber="+4934567893"/>

            <Email Remark="newsletter:yes">otto.mustermann@example.com</Email>

            <Address Remark="catalog:yes">

              <AddressLine>Musterstraße 1</AddressLine>
              <CityName>Musterstadt</CityName>
              <PostalCode>1234</PostalCode>
              <CountryName Code="DE"/>

            </Address>

          </Customer>

        </Profile>
      </ProfileInfo>
    </Profiles>
  </ResGuest>
</ResGuests>
```

sample-GuestRequests-OTA_ResRetrieveRS-book.xml - Customer element

The **Customer** element contains:

- one **PersonName** element (**mandatory**) with **NamePrefix** (**optional**), **GivenName** (**mandatory**), **Surname** (**mandatory**) and **NameTitle** (**optional**)
- one or more **Telephone** elements (either at least one **Telephone** element or the **Email** element is **mandatory**)
- one **Email** element (either at least one **Telephone** element or the **Email** element is **mandatory**)
- one **Address** element (**mandatory** for **book** requests, **optional** for **quote** requests) with **AddressLine**, **CityName**, **PostalCode** and **CountryName** with **Code** (all **mandatory**)

The **Telephone** element contains the **optional** **PhoneTechType** attribute: it indicates the phone technology (1 → voice, 3 → fax, 5 → mobile per OTA code table).

The **Email** element contains the **optional** attribute **Remark** having either values `newsletter:yes` or `newsletter:no` indicating whether the customer does or does not wish to receive an email newsletter. A missing **Remark** indicates the information is not known and should be treated the same way as if a `newsletter:no` was given (do not send a newsletter).

Analogously, the **Address** element contains the **optional** attribute **Remark** having either values `catalog:yes` or `catalog:no` indicating whether the customer does or does not wish to receive print

ads by mail. A missing **Remark** indicates the information is not known and should be treated the same way as if a `catalog:no` was given (do not send mail).

The **Code** attribute in **CountryName** **must** follow ISO 3166-1 alpha-2 (two-letter uppercase country codes).

Third part: **ResGlobalInfo**.

```
<ResGlobalInfo>

  <Comments>

    <Comment Name="included services">
      <ListItem ListItem="1" Language="de">Parkplatz</ListItem>
      <ListItem ListItem="2" Language="de">Schwimmbad</ListItem>
      <ListItem ListItem="3" Language="de">Skipass</ListItem>
    </Comment>

    <Comment Name="customer comment">
      <Text>
        Sind Hunde erlaubt?

        Mfg.
        Otto Mustermann.
      </Text>
    </Comment>

  </Comments>

  <CancelPenalties>
    <CancelPenalty>
      <PenaltyDescription>
        <Text>
          Cancellation is handled by hotel.
          Penalty is 50%, if canceled within 3 days before show, 100% otherwise.
        </Text>
      </PenaltyDescription>
    </CancelPenalty>
  </CancelPenalties>

  <HotelReservationIDs>
    <!-- ResID_Type 13 -> Internet Broker -->
    <HotelReservationID ResID_Type="13"
      ResID_Value="Slogan"
      ResID_Source="www.example.com"
      ResID_SourceContext="top banner" />
  </HotelReservationIDs>

</ResGlobalInfo>
```

sample-GuestRequests-OTA_ResRetrieveRS-book.xml - **ResGlobalInfo** element

The **ResGlobalInfo** element contains:

- **one Comment** element (**optional**) with attribute **Name** set to `included services` containing the included services given as **free text fields** using **ListItem** elements (see below). In most cases the AlpineBits client software will just display this to a human hotel employee with no further processing
- **one Comment** element (**optional**) with attribute **Name** set to `customer comment` containing a single **Text** element freely filled out by the customer and fed through unchecked by the portal

- **only allowed for book requests, one **PenaltyDescription** element (mandatory) containing a single **Text** element with no attributes that clearly states the cancellation policy the portal and the hotel have previously agreed upon and the portal has communicated to the customer - the language or languages of this text is chosen by the portal**
- **one **HotelReservationID** element (optional) that can be used for internet campaign management: **ResID_Type** must be given and equal 13 (internet broker) and the three attributes **ResID_Value**, **ResID_Source** and **ResID_SourceContext** (all optional) identify, respectively, the campaign name (Slogan in our example), the campaign source (www.example.com) and the campaign marketing medium (top banner) following the scheme used by Google Analytics**

Each **ListItem** element **must** have **Language** and **ListItem** attributes. At most **one ListItem** element is allowed for each combination of **Language** and **ListItem**.

Server Response Elements and Attributes

Name of Element / Attribute	Annotations	Mand	Rept	Children of/Attribute of
OTA_ResRetrieveRS		yes	no	- (root element)
Success		no	no	OTA_ResRetrieveRS
ReservationsList		no	no	OTA_ResRetrieveRS
HotelReservation		no	yes	ReservationsList
<i>CreateDateTime</i>		yes	no	HotelReservation
<i>ResStatus</i>	"Book" or "Quote"	yes	no	HotelReservation
UniqueID		yes	no	HotelReservation
<i>Type</i>	mandatory value: "14"	yes	no	UniqueID
<i>ID</i>		yes	no	UniqueID
RoomStays		yes	no	HotelReservation
RoomStay	please note the special case for "Quote" requests wrt. alternative periods (see text)	yes	yes	RoomStays
RoomTypes	mandatory if ResStatus="book"	*	no	RoomStay
RoomType	mandatory if ResStatus="book"	*	no	RoomTypes
<i>RoomTypeCode</i>		yes	no	RoomType
RatePlans		yes	no	RoomStay
RatePlan		yes	no	RatePlans
<i>RatePlanCode</i>		no	no	RatePlan
MealsIncluded	mandatory if ResStatus="book"	*	no	RatePlan
<i>Breakfast</i>		yes	no	MealsIncluded

<i>Lunch</i>		yes	no	MealsIncluded
<i>Dinner</i>		yes	no	MealsIncluded
<i>MealPlanCodes</i>	value "1" means "All inclusive"	no	no	MealsIncluded
GuestCounts		yes	no	RoomStay
GuestCount	at least one element must be present without "Age" attribute	*	yes	GuestCounts
<i>Age</i>		no	no	GuestCount
<i>Count</i>		yes	no	GuestCount
TimeSpan		yes	no	RoomStay
<i>Duration</i>		yes	no	TimeSpan
StartDateWindow		yes	no	TimeSpan
<i>EarliestDate</i>		yes	no	StartDateWindow
<i>LatestDate</i>		yes	no	StartDateWindow
Total	mandatory if ResStatus="book"	*	no	RoomStay
<i>AmountAfterTax</i>		yes	no	Total
<i>CurrencyCode</i>		yes	no	Total
ResGuests		yes	no	HotelReservation
ResGuest		yes	no	ResGuests
Profiles		yes	no	ResGuest
ProfileInfo		yes	no	Profiles
Profile		yes	no	ProfileInfo
Customer		yes	no	Profile
<i>Gender</i>	Values: "Male", "Female", "Unknown"	yes	no	Customer
<i>BirthDate</i>	Format: ISO 8601	no	no	Customer
<i>Language</i>	Format: ISO 639-1	yes	no	Customer
PersonName		yes	no	Customer
NamePrefix		no	no	PersonName
GivenName		yes	no	PersonName
Surname		yes	no	PersonName
NameTitle		no	no	PersonName
Telephone	either at least one of this or Email is	*	yes	Customer

	mandatory			
<i>PhoneTechType</i>		yes	no	Telephone
<i>PhoneNumber</i>		yes	no	Telephone
Email	either this or at least one Telephone is mandatory	*	no	Customer
<i>Remark</i>	value "newsletter:yes" or "newsletter:no". If missing assume "newsletter:no"	no	no	Email
Address	mandatory if ResStatus="book"	*	no	Customer
<i>Remark</i>	value "catalog:yes" or "catalog:no". If missing assume "catalog:no"	no	no	Address
AddressLine		yes	no	Address
CityName		yes	no	Address
PostalCode		yes	no	Address
CountryName		yes	no	Address
<i>Code</i>	Format: ISO 3166-1	yes	no	CountryName
ResGlobalInfo	can be empty	yes	no	HotelReservation
Comments		no	no	ResGlobalInfo
Comment		no	yes	Comments
<i>Name</i>	values: "included services" or "customer comment"	yes	no	Comment
ListItem		no	yes	Comment with name="included services"
<i>ListItem</i>		yes	no	ListItem
<i>Language</i>		yes	no	ListItem
Text		no	no	Comment with name="customer comment"
CancelPenalties	mandatory if ResStatus="book"	*	no	ResGlobalInfo
CancelPenalty	mandatory if ResStatus="book"	*	no	CancelPenalties
PenaltyDescription	mandatory if ResStatus="book"	*	no	CancelPenalty
Text	mandatory if ResStatus="book"	*	no	PenaltyDescription
HotelReservationIDs		no	no	ResGlobalInfo
HotelReservationID		no	no	HotelReservationIDs

<i>ResID_Type</i>	value: "13"	yes	no	HotelReservationID
<i>ResID_Value</i>		no	no	HotelReservationID
<i>ResID_Source</i>		no	no	HotelReservationID
<i>ResID_SourceContext</i>		no	no	HotelReservationID

Implementation tips and best practice

- If a non-standard **MealsIncluded** has to be transmitted, consider using the closest standard **MealsIncluded** combination. This needs prior agreement among the parts, which is not covered by AlpineBits. For example in South-Tyrol some hotels offer "Dreiviertel-Pension" (half board plus afternoon snack, hence a non-standard **MealsIncluded** combination) to their guests. This may be transmitted as half board, since "Dreiviertel-Pension" replaces half board for these hotels.
- The value of the **PhoneNumber** attribute (element **Telephone**) should contain the standard international format (as in +<country code><phone number>) whenever possible.

4.3. SimplePackages: Package availability notifications

When the value of the **action** parameter is `OTA_HotelRatePlanNotif:SimplePackages` the client intends to send package availability notifications to the server.

Please note that SimplePackages **do not aim** at fully describing all possible aspects and properties of a complex package, and in particular they **do not aim** at describing all properties in a fully machine processable way. Information given as free text is typically intended to be visualized on a portal, not as an input to a fully automated booking workflow (hence the name "*Simple*").

The availability of multiple packages may be transmitted within a single request, which **must** be treated in a single transaction by the server. Hence an error response means that **no data** has been accepted by the server.

Client Request (notify package availability)

The parameter **request** **must** contain an `OTA_HotelRatePlanNotifRQ` document with **one or more** **RatePlan** elements each describing an available package. The **mandatory Start** and **End** attributes indicate the start and end dates (ISO 8601) of the package's public visibility. Following is the outer part of the request document.

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_HotelRatePlanNotifRQ
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelRatePlanNotifRQ.xsd"
  Version="2.001">
  <RatePlans>
    <RatePlan Start="2012-01-01" End="2012-12-31">
      <Rates>
        <!-- availability, costs, meals, see below -->
      </Rates>
      <Description Name="title">
        <!-- see below -->
      </Description>
    </RatePlan>
  </RatePlans>
</OTA_HotelRatePlanNotifRQ>
```

```

<Description Name="intro">
  <!-- see below -->
</Description>

<Description Name="gallery">
  <!-- see below -->
</Description>

<Description Name="details">
  <!-- see below -->
</Description>

<UniqueID Type="18" ID="3f6fcef1f0da2ef57"/>

<HotelRef HotelCode="123" HotelName="Frangart Inn"/>

</RatePlan>
</RatePlans>
</OTA_HotelRatePlanNotifRQ>

```

sample-SimplePackages-OTA_HotelRatePlanNotifRQ.xml - outer part

Looking towards the end of the document first:

- the **mandatory UniqueID** element uniquely identifies the package, it **must** be of **Type 18** (meaning other per OTA code table) and the **ID must** be a 16 digit hex string; if a client sends a notification using an **UniqueID** it has sent before it wishes to update (overwrite) all information stored on the server
- the **mandatory HotelRef** element associates a hotel to each package. The attributes **HotelCode** and **HotelName** are described in room availability notifications (section 4.1)

While it is possible to send more than one **RatePlan** element, AlpineBits requires that all **RatePlan** elements refer to the same Hotel. Hence at most **one** hotel can be dealt with in a single request. An AlpineBits server **must** return an error if it receives a request referring to more than one Hotel.

The **mandatory Rates** element contains **one or more Rate** elements with data about dates, costs and meals:

```

<Rates>
<Rate MinGuestApplicable="2" Start="2012-08-01" End="2012-08-31" Sat="true" Duration="P7N">
  <BaseByGuestAmts>
    <BaseByGuestAmt NumberOfGuests="1" AmountAfterTax="499.00" CurrencyCode="EUR"/>
  </BaseByGuestAmts>
  <RateDescription Name="included services">
    <!-- included services in free text form -->
    <ListItem ListItem="1" Language="en">parking lot (included)</ListItem>
    <ListItem ListItem="2" Language="en">swimming pool (included)</ListItem>
    <ListItem ListItem="3" Language="en">full board (+25 EUR)</ListItem>
    <ListItem ListItem="1" Language="it">parcheggio</ListItem>
    <ListItem ListItem="2" Language="it">piscina</ListItem>
    <ListItem ListItem="3" Language="it">pensione completa (+25 EUR)</ListItem>
    <ListItem ListItem="1" Language="de">Parkplatz</ListItem>
    <ListItem ListItem="2" Language="de">Schwimmbad</ListItem>
    <ListItem ListItem="3" Language="de">Vollpension (+25 EUR)</ListItem>
  </RateDescription>
  <MealsIncluded Breakfast="true" Lunch="false" Dinner="true"/>
</Rate>
<Rate MinGuestApplicable="2" Start="2012-12-01" End="2012-12-31" Sat="true" Duration="P7N">

```

```

<BaseByGuestAmts>
  <BaseByGuestAmt NumberOfGuests="1" AmountAfterTax="599.00" CurrencyCode="EUR"/>
</BaseByGuestAmts>

<RateDescription Name="included services">
  <!-- included services in free text form -->
  <ListItem ListItem="1" Language="en">parking lot</ListItem>
  <ListItem ListItem="2" Language="en">skipass</ListItem>
  <ListItem ListItem="1" Language="it">parcheggio</ListItem>
  <ListItem ListItem="2" Language="it">skipass</ListItem>
  <ListItem ListItem="1" Language="de">Parkplatz</ListItem>
  <ListItem ListItem="2" Language="de">Skipass</ListItem>
</RateDescription>

<MealsIncluded Breakfast="true" Lunch="true" Dinner="true"/>

</Rate>

</Rates>

```

sample-SimplePackages-OTA_HotelRatePlanNotifRQ.xml - **Rate** elements

In the example the first of the two **Rate** elements means that the package is 7 nights with arrival date any Saturday in August 2012 (attributes **Start** and **End** are **mandatory**, for optional **Sat** attribute and more information on DOW limits see section 4.1). The **Duration** attribute is **mandatory** and is encoded in ISO 8601. AlpineBits allows only durations given in nights, the form is thus always P×N where x is a number.

Total cost is 499 EUR (expressed by **mandatory** attributes **AmountAfterTax** and **CurrencyCode**) per person (**mandatory** attribute **NumberOfGuests**) with a minimum of 2 persons (**mandatory** attribute **MinGuestApplicable**). AlpineBits requires **NumberOfGuests** to be always 1. That means package prices are always per person. Also, package prices are always to be considered starting prices: the 499 EUR in this example should be displayed to the customer as "starting from 499 EUR".

The **mandatory** element **MealsIncluded** has attributes **Breakfast**, **Lunch** and **Dinner** set to true or false - all three **must** be given, even if their value is false; the **optional** attribute **MealPlanCodes** can be used as described in section 4.2, and is not shown in the example.

The included services are given as **free text fields** in the optional **RateDescription** element using **ListItem** elements. Each **ListItem** element **must** have **Language** and **ListItem** attributes. At most **one** **ListItem** element is allowed for each combination of **Language** and **ListItem**. These are intended for display to humans, not for automated processing.

The second **Rate** elements specifies another period of validity of the same package: the example package can be booked also in december, it has full board then, includes a skipass instead of the swimming pool and costs 599 EUR.

Following are four **Description** elements.

```

<!-- the title of the package (plain text),
      repeated for each language -->

<Description Name="title">
  <Text TextFormat="PlainText" Language="en">Hiking in the Atacama Desert</Text>
  <Text TextFormat="PlainText" Language="it">Escursione nel deserto di Atacama</Text>
  <Text TextFormat="PlainText" Language="de">Bergwandern in der Atacamawüste</Text>
</Description>

<!-- the short introductory text (plain text) and optional URLs,
      repeated for each language -->

<Description Name="intro">

  <Text TextFormat="PlainText" Language="en">

```

```

    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
    incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
</Text>
<URL>http://www.alpinebits.org/en/</URL>

<Text TextFormat="PlainText" Language="it">
    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
    incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
</Text>
<URL>http://www.alpinebits.org/it/</URL>

<Text TextFormat="PlainText" Language="de">
    Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
    incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
    exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
</Text>
<URL>http://www.alpinebits.org/de/</URL>

</Description>
<!-- the images associated with the package: copyright/caption, image and optional URLs,
    repeated for each language -->

<Description Name="gallery">

    <Text TextFormat="PlainText" Language="en">
        (C) 2012 Example Inc.
    </Text>
    <Image>http://www.example.com/image-en.gif</Image>
    <URL>http://www.example.com/en/</URL>

    <Text TextFormat="PlainText" Language="it">
        (C) 2012 Example Inc.
    </Text>
    <Image>http://www.example.com/image-it.gif</Image>
    <URL>http://www.example.com/it/</URL>

    <Text TextFormat="PlainText" Language="de">
        (C) 2012 Example Inc.
    </Text>
    <Image>http://www.example.com/image-de.gif</Image>
    <URL>http://www.example.com/de/</URL>

</Description>

<!-- the detail description as texts (plain text, HTML) or ListItems,
    repeated for each language -->

<Description Name="details">

    <Text TextFormat="PlainText" Language="en">
        Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
        incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
        exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
    </Text>
    <Text TextFormat="HTML" Language="en">
        <![CDATA[
            Duis aute <b>irure dolor</b> in reprehenderit involuptate velit esse cillum dolore
            eu <a href="http://www.alpinebits.org/">fugiat nulla pariat</a>. Excepteur sint
            occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim
            id est laborum.
        ]]>
    </Text>
    <ListItem ListItem="1" Language="en">commodo consequat</ListItem>
    <ListItem ListItem="2" Language="en">voluptate velit</ListItem>

    <!-- other languages not shown... -->

</Description>

```

Title.

The first **Description** element has **Name** set to `title` and is **mandatory**. This is the title of the package.

It **must** contain nothing but **Text** elements (at least one). Each **Text** element **must** have **TextFormat** set to `PlainText` and the **Language** attribute **must** be given.

At **most** one **Text** element per language is allowed.

Intro.

The second **Description** element has **Name** set to `intro` and is mandatory as well. This is the short introductory text of the package with optional URLs.

It must contain one or more **Text** elements. Each **Text** element must have **TextFormat** set to `PlainText` and the **Language** attribute must be given. Each **Text** element can be followed by zero or more **URL** elements.

The **URL** elements are implicitly associated with the text (and the language of the text) that precedes them.

At **most** one **Text** element per language is allowed.

Gallery.

The third **Description** element with **Name** set to `gallery` is **optional**. It contains **one or more** images associated with the package.

For each image there is a **Text** element with the **TextFormat** set to `PlainText`. It describes the caption/copyright for the image. The **Language** attribute is mandatory.

Following the **Text** element is the **mandatory Image** element containing the image location (HTTP URL). The **Image** element can be followed by zero or more **URL** elements.

The **Image** and **URL** elements are implicitly associated with the caption/copyright text and the language that precedes them. In the present example there are 3 images.

Of course, there can be more than one **Image** per language.

Details.

The fourth and last **Description** element with **Name** set to `details` is **optional**, it can contain **zero or more Text** elements and **zero or more ListItem** elements. It **must** contain **at least one** element, however.

Each **Text** element **must** have **TextFormat** set to `PlainText` or `HTML`, and the **Language** attribute **must** be given. At most **one Text** element is allowed for each combination of **Language** and **TextFormat**. The presence of a **Text** element with **TextFormat** set to `HTML` is intended as rich text alternative of a **Text** element with **TextFormat** set to `PlainText` of the same **Language** and makes the latter **mandatory**.

Please note that an AlpineBits server is explicitly allowed to **filter, shorten or even skip the HTML content**, therefore the usage of **Text** elements with **TextFormat** set to `HTML` is **not** recommended but left as an option for implementers that absolutely need it.

The **ListItem** elements are intended to provide a structured description of the package that will be shown to the end user. The appearance of the structured description depends on the server implementation, it is not guaranteed to follow the textual description. Each **ListItem** element **must**

have **Language** and **ListItem** attributes. At most **one ListItem** element is allowed for each combination of **Language** and **ListItem**.

Client Request (notify that a package is no longer available)

A client that wishes to notify the server that a package is no longer available it will send **one or more RatePlan** elements that only contain a **UniqueID** element and are otherwise empty such as shown in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<OTA_HotelRatePlanNotifRQ
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelRatePlanNotifRQ.xsd"
  Version="2.001">

  <RatePlans>

    <RatePlan>

      <UniqueID Type="18" ID="3f6fcef1f0da2ef57"/>

    </RatePlan>

  </RatePlans>

</OTA_HotelRatePlanNotifRQ>
```

sample-SimplePackages-OTA_HotelRatePlanNotifRQ-remove.xml

A server must return an error response if a client tries to notify that a package not in the server records (**ID**) is no longer available.

All given **RatePlan** elements must refer to the same hotel: at most **one** hotel can be dealt with in a single request. An AlpineBits server **must** return an error if it receives a request referring to more than one Hotel.

Please note that it is **not** allowed to mix notifications that packages are available with notifications that packages are no longer available in the same request.

Client Request Elements and Attributes

Name of Element / <i>Attribute</i>	Annotations	Mand	Rept	Children of/Attribute of
OTA_HotelRatePlanNotifRQ		yes	no	- (root element)
RatePlans		yes	no	OTA_HotelRatePlanNotifRQ
RatePlan		yes	yes	RatePlans
<i>Start</i>		yes	no	RatePlan
<i>End</i>		yes	no	RatePlan
Rates		yes	no	RatePlan
Rate		yes	yes	Rates
<i>MinGuestApplicable</i>		yes	no	Rate

<i>Start</i>	Format: ISO 8601	yes	no	Rate
<i>End</i>	Format: ISO 8601	yes	no	Rate
<i>Duration</i>	Format: ISO 8601	yes	no	Rate
<i>Mon</i>	"true" or "false"	no	no	Rate
<i>Tue</i>	"true" or "false"	no	no	Rate
<i>Weds</i>	"true" or "false"	no	no	Rate
<i>Thur</i>	"true" or "false"	no	no	Rate
<i>Fri</i>	"true" or "false"	no	no	Rate
<i>Sat</i>	"true" or "false"	no	no	Rate
<i>Sun</i>	"true" or "false"	no	no	Rate
BaseByGuestAmts		yes	no	Rate
BaseByGuestAmt		yes	no	BaseByGuestAmts
<i>NumberOfGuests</i>	mandatory value: "1"	yes	no	BaseByGuestAmt
<i>AmountAfterTax</i>		yes	no	BaseByGuestAmt
<i>CurrencyCode</i>		yes	no	BaseByGuestAmt
RateDescription		no	no	Rate
<i>Name</i>	mandatory value: "included services"	yes	no	RateDescription
ListItem		no	yes	RateDescription
<i>ListItem</i>		yes	no	ListItem
<i>Language</i>		yes	no	ListItem
MealsIncluded		yes	no	Rate
<i>Breakfast</i>		yes	no	MealsIncluded
<i>Lunch</i>		yes	no	MealsIncluded
<i>Dinner</i>		yes	no	MealsIncluded
<i>MealPlanCodes</i>	value "1" means "All inclusive"	no	no	MealsIncluded
Description	mandatory: one element with Name="title", one element with Name="intro", zero or one elements with Name="gallery", zero or one element with Name="details"	*	yes	RatePlan
<i>Name</i>	allowed values: "title", "intro", "gallery", "details"	yes	no	Description
Text		yes	yes	Description

<i>TextFormat</i>	allowed values: "Plaintext", "HTML" only if parent Description has Name="details"	yes	no	Text
<i>Language</i>		yes	no	Text
URL		no	yes	Description with Name="intro" or Name="gallery"
Image		yes	yes	Description with Name="gallery"
ListItem		no	yes	Description with Name="details"
<i>ListItem</i>		yes	no	ListItem
<i>Language</i>		yes	no	ListItem
UniqueID		yes	no	RatePlan
<i>Type</i>	mandatory value: "18"	yes	no	UniqueID
<i>ID</i>	format: 16 hex digits	yes	no	UniqueID
HotelRef		yes	no	RatePlan
<i>HotelCode</i>	either this or HotelName are mandatory	*	no	HotelRef
<i>HotelName</i>	either this or HotelCode are mandatory	*	no	HotelRef

Server response

The server will send a response indicating the outcome of the request. The response is a `OTA_HotelRatePlanNotifRS` document. Three types of outcome are possible: success, warning or error.

Success

The request was accepted and processed successfully. The client does **not** need to take any further action. In this case the `OTA_HotelRatePlanNotifRS` response contains nothing but a **single**, empty **Success** element:

```
<?xml version="1.0" encoding="UTF-8"?>

<OTA_HotelRatePlanNotifRS
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelRatePlanNotifRS.xsd"
  Version="3.14">

  <Success/>

</OTA_HotelRatePlanNotifRS>
```

`sample-SimplePackages-OTA_HotelRatePlanNotifRS-success.xml`

Warning

The request was accepted and processed successfully. However, one or more non-fatal problems were detected and added to the server response. The client does **not** need to resend the request, but **must** notify the user or the client implementer regarding the warnings received.

In this case the OTA_HotelRatePlanNotifRS response contains an empty **Success** element followed by **one or more Warning** elements with the attribute **Type** set to the fixed value 11 ("Advisory" according to OTA) containing a human readable text.

If a **Warning** element is not regarding the request as a whole but specific to a **RatePlan** element in the request, the **Warning** **must** also have a **RecordID** attribute.

The value of **RecordID** must be the value of the **RatePlan/UniqueID/ID** the **Warning** is referred to.

Here is an example warning:

```
<OTA_HotelRatePlanNotifRS
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelRatePlanNotifRS.xsd"
  Version="3.14">

  <Success/>
  <Warnings>
    <Warning Type="11" RecordID="3f6fcef0da2ef57">
      Warning: End date is less than 3 days from now.
    </Warning>
  </Warnings>

</OTA_HotelRatePlanNotifRS>
```

sample-SimplePackages-OTA_HotelRatePlanNotifRS-warning.xml

Error

The request **could not** be accepted or processed successfully. The client **must** take action: it **may** try to resend the message if it has reason to assume the error is transient and occurred for the first time, otherwise it **must** escalate the problem to the user or client implementer.

In this case the OTA_HotelAvailNotifRS response contains **one or more Error** elements each with a **Type** attribute containing a human readable text.

```
<OTA_HotelRatePlanNotifRS
  xmlns="http://www.opentravel.org/OTA/2003/05"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_HotelRatePlanNotifRS.xsd"
  Version="3.14">

  <Errors>
    <Error Type="404">
      Invalid start/end date combination
    </Error>
  </Errors>

</OTA_HotelRatePlanNotifRS>
```

sample-SimplePackages-OTA_HotelRatePlanNotifRS-error.xml

Exactly what errors a server will trap is implementation-specific, but it goes without saying that a server **must** return an error if it cannot process any part of a message.

The value of the type attribute **must** follow the OTA2010A error codes table that comes with the OTA2010A documentation package. The package can be downloaded from the OTA web site ³. The file `OpenTravel_CodeList_20100322.xls` contains the error codes table.

Implementors are **required** to follow that table. Here is a non-exhaustive list of error types that might be encountered when processing SimplePackage requests, taken from the OTA table:

Type	Text
61	Invalid hotel
392	Invalid hotel code
396	Invalid name
375	Hotel not active
135	End date is invalid
136	Start date is invalid
145	Duration period or dates incorrect
404	Invalid start/end date combination
407	Item too long
425	No match found

Server Response Elements and Attributes

Name of Element / Attribute	Annotations	Mand	Rept	Children of/Attribute of
OTA_HotelRatePlanNotifRS		yes	no	- (root element)
Success		no	no	OTA_HotelRatePlanNotifRS
Warnings		no	no	OTA_HotelRatePlanNotifRS
Warning		no	yes	Warnings
Type	value = "11"	yes	no	Warning
RecordID	mandatory if the warning refers to a specific RatePlan	*	no	Warning
Errors		no	no	OTA_HotelRatePlanNotifRS
Error		no	yes	Errors
Type	OTA error codes	yes	no	Error

Implementation tips and best practice

- If a non-standard **MealsIncluded** has to be transmitted, consider using the closest standard **MealsIncluded** combination. This needs prior agreement among the parts, which is not covered by AlpineBits. For example in South-Tyrol some hotels offer "Dreiviertel-Pension" (half board plus afternoon snack, hence a non-standard **MealsIncluded** combination) to their guests. This may be transmitted as half board, since "Dreiviertel-Pension" replaces half board for these hotels.

A. Example code snippets

A public repository with example code snippets is online at <https://github.com/alpinebits/code-snippets>. Contributions are welcome (any programming language).

B. Protocol Version Compatibility

2012-05b client talking to 2013-04 server

Housekeeping

The client will not send the X-AlpineBits-ClientID field in the HTTP header, since it is not aware of this feature. This will cause authentication problems with those 2013-04 servers that require an ID.

The client will not send the X-AlpineBits-ClientProtocolVersion field in the HTTP header, since it is not aware of this feature. This is no problem: a server that is interested in this, will simply recognize the client as preceding protocol version 2013-04.

If the client checks the server version it will see 2013-04 - a version it doesn't recognize. Likewise, if the client checks the capabilities it might see the `OTA_HotelAvailNotif_accept_deltas` capability. Client implementers interested to have their 2012-05b client talk to a 2013-04 server should verify this is not a problem for their client software.

FreeRooms

There is no compatibility problem in the request part: the client will not send partial information (deltas), since it is simply not aware of the existence of the feature.

Please be aware that the lack of this feature (obviously) causes more data to be sent to the server, something not all companies that run servers will be happy with.

The server response might contain a **Warning** element the client cannot process. If the client - as it should - carefully parses the response, it will treat this as an error situation and act accordingly. So basically the client is expected to treat the warnings as error, which might be an issue and should be tested for.

GuestRequests

No compatibility problems are expected.

SimplePackages

2013-04 introduced the limitation that all packages sent within a single request must refer to the same hotel. A older client not aware of this limit might incur an error returned from the server error.

Similar to the FreeRooms case, the server response might contain a **Warning** element the client cannot process. If the client - as it should - carefully parses the response, it will treat this as an error situation and act accordingly. So basically the client is expected to treat the warnings as error, which might be an issue and should be tested for.

2013-04 client talking to 2012-05b server

Housekeeping

The client sends the X-AlpineBits-ClientProtocolVersion field and may send the X-AlpineBits-ClientID field in the HTTP header, but the server will just ignore it - being unaware of the feature.

If the client checks the server version and/or checks the capabilities - as it should - it will note the missing features and not use them.

Hence, technically there is no problem with this combination.

FreeRooms

The client will not send partial information (deltas), since the server does not export the `OTA_HotelAvailNotif_accept_deltas` capability.

The server will never send a response with a **Warning** element. This is not a problem for the client.

GuestRequests

In 2013-04 the form of the **ID** attribute is not any more restricted. It has become a free text field. An older server might insist on the old form and throw an error.

SimplePackages

The server will never send a response with a **Warning** element. This is not a problem for the client.

C. Links

[0] **Creative Commons BY ND license:**

<http://creativecommons.org/licenses/by-nd/3.0/>

[1] **HTTP basic access authentication:**

http://en.wikipedia.org/wiki/Basic_access_authentication/

[2] **OpenTravel Alliance:**

<http://www.opentravel.org/>

[3] **OTA2010A documentation package download:**

<http://www.opentravel.org/Specifications/ReleaseNotes.aspx?spec=2010A>

[4] **OTA2010A XML schema files online:**

<http://www.opentravel.org/Specifications/SchemaIndex.aspx?FolderName=2010A>

[5] **browsable interface to the above schema files:**

<http://adriatic.pilotfish-net.com/ota-modelviewer/>