NetSuite OpenAir SOAP API Reference Guide

January 19, 2013

NETSUITE | OpenAir

NetSuite OpenAir SOAP API Reference Guide

January 3, 2013

Trademarks

Contents

Chapter 4 Web Services Method Complex Types

Chapter 5 Custom Fields

Chapter 6 OpenAir Complex Types

Chapter 7 Setting Application Switches Via the API

Chapter 8 Code Examples

Appendix A Error Code Listing

Appendix B OpenAir Data Dictionary

Appendix C Best Practices

Troubleshooting

New Features

Chapter 1 Introduction to OpenAir Web Services

NetSuite OpenAir provides OpenAir Web Services as a layer for the exchange of NetSuite OpenAir data between the main site and peripheral programs. These programs include partnered Web sites, OpenAir in-house applications that don't need direct database access, and third party applications indirectly supported through OpenAir. Before you begin using this service, we recommend that you review Appendix C Best Practices.

The application programming interface (API) is data-centric, but it is not a direct line into the OpenAir database. While it provides access to much of the information on OpenAir, it is a layer of indirection from the actual database structure. OpenAir's database structure may change, but applications that use the API will not need to change. OpenAir Web Services exist in addition to the OpenAir XML API and serve as a wrapper around the XML API, providing the same or very similar functionality. See OpenAir XML API Reference Guide for detailed documentation.

Technology

OpenAir Web Services are based on Simple Object Access Protocol (SOAP), an XML-based convention. The following standards are observed.

Standard Name	Web Site Reference
Simple Object Access Protocol (SOAP) 1.1	http://www.w3.org/TR/2000/NOTE-SOAP-20000508/
Web Service Description Language (WSDL) 1.1	http://www.w3.org/TR/2001/NOTE-wsdl-20010315

Since OpenAir Web Services are database-driven, it is important that the information you collect from your users is compatible with the fields in the database. We provide you with the list of commands and data types that are meaningful to an OpenAir database. Through the use of commands and data types provided by the API, and by limiting the values that can be stored in each data type, your data will always be consistent with, and able to be stored within, an OpenAir database.

Target Audience

This document is intended for developers of applications that will connect to the OpenAir Web site.

Overview

The following provides a brief description of what's included in OpenAir SOAP Web Services.

- **Getting Started** — includes steps for what you should do to begin using OpenAir Web Services. It includes general procedures for connecting to the OpenAir API. It also includes specific instructions for integrations based on Apache Axis libraries or .NET Framework with Microsoft Visual Studio as the IDE.
- **Methods** — lists the supported calls in the API and the syntax, use, arguments, and response of each call as well as sample C# code and Java code.
- **Web Services Method Complex Types** - lists OpenAir-specific datatypes that are required for executing requests and reading responses with the SOAP API. Provides datatype properties and methods in which the specific type is used.
- **Custom Fields** — introduces custom fields and provides information for reading custom field values, requesting custom fields for an object, and modifying records for custom field values.
- **OpenAir Complex Types** — lists OpenAir-specific datatypes defining the business object model exposed via the SOAP API. Provides properties and supported methods for each datatype.
- **Setting Application Switches Via the API** — describes company and user-level switches you can set using the API.
- **Code Examples** — provides code examples for specific functions such as creating multiple users.
- **Appendix A - Error Code Listing** — identifies common errors and associated codes.
- **Appendix B - OpenAir Data Dictionary** — provides a reference to the OpenAir Data Dictionary: https:www.openair.com/database/single_user.html
- **Appendix C - Best Practices —** provides a guide for preparing for and using the API.

Definitions

The following are definitions used in OpenAir SOAP Web Services.

- **XML** — eXtensible Markup Language
- **API** — Application Program Interface
- **Server** — The OpenAir site that understands the API
- **Client** — Application that talks to Server using the API
- **XML structure** — An XML element that contains other XML elements
- **OA** — Abbreviation for OpenAir
- **SOAP** — Simple Object Access Protocol

- **XSD** — XML Schemas specification
- **WSDL** — Web Service Description Language

Error Handling

There are two types of errors returned by the API. They are SOAP Fault Errors and API Method Level Errors. Each is described as follows.

- **SOAP Fault Errors** — are returned in the following circumstances: incorrect usage, badly formatted SOAP messages, and failed authentication. They result in SOAP Fault messages with a specific error code and often a string description. When a string description is missing, the read method can be used to retrieve the specific description for an error that occurred. (For more information, refer to the read method.)
- **API Method Level Errors** —Errors specific to a query or action are returned as a collection of oaError objects. See Appendix A Error Code Listing.

  Note: String errors are not guaranteed to be returned. If oaError object's string is empty, look up the error code description by issuing a read method for the "Error" object.

Date Format

Most date fields use the following date format: YYYY-MM-DD HH:MM:SS

Limits

Currently there are four types of usage limits that are enforced in OpenAir Web Services.

- There is a limit of 1000 records that can be requested at one time. Use the limit attribute on the read method ReadRequest argument to limit the amount of records being returned and request records in batches. If read is used without the "limit" attribute, an error will be returned.
- There is a limit of 1000 objects any method can accept, so if you need to use add, upsert, modify, createUser, or submit methods, make sure to load the records in batches. The server will return an error if more objects are specified.
- There is a frequency limit of daily transactions allowed for each account.
- There is a frequency limit of transactions allowed for each 60-second interval for each account.

Please contact your Account Manager with any questions about the frequency limits. After a frequency limit is reached for either daily transactions or a 60-second interval, our web servers will respond with 403 "access denied" error until the end of the period. The best way to avoid breaching these limits is to make sure all records and fields are requested by batching many commands into one request call. Also, avoid making any requests within a loop. See Appendix C Best Practices.

Chapter 2 Getting Started

The following are instructions for what you should do to begin using OpenAir Web Services.

How to Begin

Step 1: Obtain OpenAir API Access

Contact the OpenAir Support Department or your account representative to request API access. See Troubleshooting for instructions. When access is granted, you will receive an API namespace and an API key. These are the two pieces of information required for API access in addition to your regular OpenAir login credentials.

The namespace and key attributes are used to verify that the request is coming from a valid partner that has permission to use our API. You will not be able to access an account with just the namespace and the key. You will also need to know the Company ID, User ID, and Password of the account.

Step 2: Generate the OpenAir Web Service WSDL

Point your browser or development environment to the following URL and the WSDL file will be automatically generated: http://www.openair.com/wsdl.pl?wsdl

Replace the server name with name of the server you're using for testing or production if it is different than our production server. **We strongly recommend that you use a test account on one of our testing servers before running in production!**

Step 3: Import the WSDL File into Your Development Platform

After you have generated the WSDL file, you import it into your development platform. Refer to Instructions for Apache Axis Libraries and Instructions for Microsoft Visual Studio.

Step 4: Set up an OpenAir Service URL

Configure your integration to connect to an OpenAir SOAP API service at one of the following urls:

OpenAir SOAP API Service Point	Service information
sandbox	https://sandbox.openair.com/soap
production	https://www.openair.com/soap

Instructions for Apache Axis Libraries

Note: OpenAir Web Services has been tested and is verified to be compatible with versions 1.3 and 1.4 of the Apache Axis SOAP libraries. It is **NOT** compatible with the Axis2 libraries.

Java client-binding objects can be used to access the API. They function as proxies for server-side equivalents. You generate these objects from your WSDL file before you can begin using the API. Ensure the following:

- You have generated the Java objects from the OpenAir WSDL file.
- For Apache Axis, you are using the WSDL2Java utility. Refer to: http://ws.apache.org/axis/java/user-guide.html#WSDL2JavaBuildingStubsSkeletonsAndDataTypesFromWSDL
- You have previously installed Axis on your system. The JAR files need to be referenced in your classpath. For more information and to obtain a copy of the Axis libraries and tools, refer to http://ws.apache.org/axis.

To proceed, refer to the following information:

- Syntax for WSDL2Java:
  java -classpath path to JAR filename `org.apache.axis.wsdl.WSDL2Java -a wsdlURL`
- -a switch generates code for elements whether they are referenced or not. For more information, refer to http://ws.apache.org/axis/java/reference.html
- JAR files in more than one location should have a semicolon separating them.

  For example, if the Axis 1.4 JAR files are installed in `C:\axis-1_4`, and the WSDL is located at `http://www.openair.com/wsdl.pl?wsdl`, the following command generates all needed stub objects and proxies to access OpenAir Web Services:

```
java -cp c:\axis-1_4\lib\axis.jar;
c:\axis-1_4\lib\axis-ant.jar;
c:\axis-1_4\lib\commons-logging-1.0.4.jar;
c:\axis-1_4\lib\commons-discovery-0.2.jar;
c:\axis-1_4\lib\jaxrpc.jar;
c:\axis-1_4\lib\log4j-1.2.8.jar;
c:\axis-1_4\lib\wsdl4j-1.5.1.jar;
org.apache.axis.wsdl.WSDL2Java -a http://www.openair.com/
wsdl.pl?wsdl
```

  A set of folders and Java source code files are generated and placed in the same directory from which the command is run. Once compiled, you can include them when you are creating your client application.

Also note that adding the -p switch will allow you to specify your own package namespace instead of the default namespace of com.soaplite.namespaces.perl.

Wizard-based tools can be used instead of the command line for this process in most Java development environments.

Periodically, additional fields are added to the OpenAir WSDL document and associated SOAP objects. These changes may not be automatically reflected in stubs generated using the Axis WSDL2Java tool and may cause an exception in client code. To handle this situation, the following workaround can be used:

Create a subclass of org.apache.axis.encoding.ser.BeanDeserializer and override the onStartChild method to handle any SAXExceptions that may occur:

```
public class MyDeserializer extends BeanDeserializer {
      public MyDeserializer(java.lang.Class javaType,
       javax.xml.namespace.QNamexmlType,org.apache.axis.description.TypeDesc
       typeDesc)
      {
       super(javaType, xmlType, typeDesc);
      }

      public SOAPHandler onStartChild(String arg0, String arg1, String
       arg2, Attributes arg3, DeserializationContext arg4) throws
       SAXException
       {
            // TODO Auto-generated method stub
            try{
                 __return super.onStartChild(arg0, arg1, arg2,arg3, arg4);
            }catch (SAXException e){
                 __return null;
            }
      }
}
```

For each generated client class (i.e., OaEnvelope, OaTimesheet, etc.), change the getDeserializer method to use the new subclass of BeanDeserializer instead of the default implementation:

```
       /**
        * Get Custom Deserializer
        **/
       public static org.apache.axis.encoding.Deserializer getDeserializer(
              java.lang.String mechType,
              java.lang.Class _javaType,
              javax.xml.namespace.QName _xmlType) {
          return
            new  MyDeserializer(
              _javaType, _xmlTpe, typeDesc);
        }
```

Instructions for Microsoft Visual Studio

Visual Studio languages use classes to access the API. Objects generated from these classes definitions function as proxies for their server-side equivalents. You must generate these classes from the OpenAir WSDL file before you can begin using the API. Proxy classes can be created using Visual Studio IDE or a command-line utility. Proxy classes can be added to a new or an existing project opened in Visual Studio.

An XML client is an application that talks to the OpenAir server using the OpenAir SOAP Web service. That client may be a Web or Windows Forms application, or even another Web service. When you access OpenAir Web Services using SOAP, infrastructure code is managed by proxy classes and the .NET Framework.

To access OpenAir Web Services, begin by adding a Web reference, identifying the namespace, creating an instance of the proxy class, and accessing methods of the class. For more information on supported calls in the API, refer to Methods.

To Add a Web Reference:

1. On the Project menu, select Add Web Reference.
2. In the Add Web Reference dialog box, type the URL:
   http://www.openair.com/wsdl.pl?wsdl
3. Click Go.
   Information about the OpenAir Web Services displays.
4. Enter "OA" in Web Reference name edit box.
5. Click Add Reference.
   A Web reference is added and a proxy classes are generated that interfaces between your application and OpenAir Web Services.

   Note: For more information about adding a Web reference, refer to "Adding and Removing Web References" in the Microsoft Visual Studio documentation.

Chapter 3 Methods

The following are supported calls in the API. Click on a call name to see the syntax, use, and code samples for that call.

Operation	Description
login	Use this function to authenticate. It returns a valid sessionId which can be used for successive calls.
version	Use this function to request the current version of an application.
read	Use this function to read data from OpenAir.
add	Use this function to add data to OpenAir.
upsert	Use this function to add or modify data to OpenAir based on a lookup attribute.
createAccount	Use this function to create OpenAir accounts.
createUser	Use this function to create OpenAir users.
submit	Use this function to submit OpenAir entities for approval.
makeURL	Use this function to create a valid URL to an OpenAir page specified.
modify	Use this function to authenticate. It returns a valid sessionId which can be used for successive calls.
delete	Use this function to request the current version of an application.
whoami	Use this function to read data from OpenAir.
servertime	Use this function to add data to OpenAir.
logout	Use this function to add or modify data to OpenAir based on a lookup attribute.

login

Syntax

```
LoginResult lr = _svc.login(loginParams);
```

Use

The login call is used to initiate a session which can be used for making successive calls to the API. sessionId which is returned as part of the LoginResult object is the unique identifier for this user's session. It can be invalidated by calling logout.

Arguments

Name	Type	Description
login	LoginParams	LoginParam object

Response

LoginResult

Sample Code - C#

```
// Create service stub
OAirServiceHandlerService _svc = new OAirServiceHandlerService();

// create LoginParam object
LoginParams loginParams = new LoginParams();
loginParams.api_namespace = "my namespace";
loginParams.api_key = "********";
loginParams.company = "company name";
loginParams.user = "username";
loginParams.password = "password";
loginParams.client = "my client name";
loginParams.version = "1.0";
LoginResult loginResult = _svc.login(loginParams);

// Create a new session header object
// Add the session ID returned from the login
_svc.SessionHeaderValue = new SessionHeader();
_svc.SessionHeaderValue.sessionId = loginResult.sessionId;
```

Sample Code --Java

```
// create our login parameters
LoginParams lp = new LoginParams();
lp.setUser("username");
lp.setPassword("password");
lp.setCompany("company name");
lp.setApi_namespace("my namespace");
lp.setApi_key("********");
lp.setClient("my client name");
lp.setVersion("1.0");

// set the service URL from our arguments
OAirServiceHandlerServiceLocator locator = new
OAirServiceHandlerServiceLocator();
locator.setOAirServiceAddress("https://www.openair.com/soap");
```

```
// now login
OAirServiceSoapBindingStub binding =
(OAirServiceSoapBindingStub)locator.getOAirService();
LoginResult loginResult = binding.login(lp);

// Create a new session header object
// Add the session ID returned from the login
SOAPHeaderElement header = new SOAPHeaderElement("http://
www.openair.com/OAirService", "SessionHeader");
SOAPElement node = header.addChildElement("sessionId");
node.addTextNode(loginResult.getSessionId());
binding.setHeader(header);
```

version

Syntax

```
VersionResult version = stub.version("My app", "1.1");
```

Use

The version call is used to request the current version of a thin client application supported by Openair.

Arguments

Name	Type	Description
name	string	Name of the application.
number	string	Version number.

Response

VersionResult

Sample Code - C#

```
VersionResult version = stub.version("My app", "1.1");
```

Sample Code - Java

```
VersionResult version = binding.version("My app", "1.1");
```

read

Syntax

```
ReadResult[] results = stub.read(new ReadRequest[2] {read1, read2});
```

Use

Use read command to retrieve data from OpenAir. Read command accepts an array of ReadRequest objects as a parameter and returns an array of corresponding ReadResult objects. Parameter specification is discussed in detail in the ReadRequest section.

Arguments

Name	Type	Description
read	[] ReadRequest	Array of ReadRequest objects

Response

```
Array of ReadResult objects
```

C# Read Code Examples

Note that "limit" attribute is **always required**, but for the sake of saving space, only the first example shows the loop which correctly gets multiple batches of data.

Example I. read equal to C#

Read the fields id, nickname, updated for users with nickname 'jsmith'.

```
ReadResult[] results;
ReadRequest rr = new ReadRequest();
rr.type = "User";
rr.method = "equal to"; //return only records that match search
criteria
rr.fields = "id, nickname, updated"; //specify fields to be returned.

//Specify search criteria
oaUser user = new oaUser();
user.nickname = "jsmith";
rr.objects = new oaBase[1] { user }; //pass in one object with search
criteria


int index = 0;    //Starting index
const int LIMIT = 1000; //Return maximum of 1000 records per request
```

```
do
{
    // Limit attribute is required.
    OA.Attribute attr = new OA.Attribute();
    attr.name = "limit";
    attr.value = String.Format("{0}, {1}", index, LIMIT);

    rr.attributes = new OA.Attribute[] { attr };
    results = _svc.read(new ReadRequest[] { rr });

    if (results != null && results.Length > 0 && results[0].errors !=
null)
    {
     foreach (oaError err in results[0].errors)
     {
       Debug.WriteLine(string.Format("Error {0} - {1}", err.code,
err.text));
     }
    }

    // get next 1000 records
    index += LIMIT;

} while (results[0].objects != null && results[0].objects.Length >
0);
```

Example II. read not equal to C#

Read id, nickname, updated fields for users that do not match certain search criteria. For more information, see oaFieldAttribute.

```
ReadRequest rr = new ReadRequest();
rr.type = "User";
rr.method = "not equal to"; //return only records that do not match
search criteria
rr.fields = "id, nickname, updated"; //specify fields to be returned

oaUser user = new oaUser();
user.nickname = "jsmith";
rr.objects = new oaBase[1] { user }; //pass in one object with search
criteria

// Limit attribute is required.
OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
```

```
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

ReadResult[] results = _svc.read(new ReadRequest[1] { rr });
```

Example III. read custom field definitions C#

Read all custom fields associated with project record type.

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "CustField";

oaCustField cf = new oaCustField();
cf.association = "project"; // custom field association

// Limit attribute is required.
OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

rr.objects = new oaBase[] { cf };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example IV. read custom field values C#

Read custom field values for a given project.

```
ReadRequest rr = new ReadRequest();
rr.method = "custom equal to"; //all custom fields associated with the
project are returned
rr.type = "Project"; //See table of objects that have custom field
support

oaCustomField cf = new oaCustomField();
cf.id = "238"; // ID of the project
rr.objects = new oaBase[] { cf };

OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

ReadResult[] results = _svc.read(new ReadRequest[] { rr }); //
returns name/value pairs.
```

Note: Review the chapter that addresses Custom Fields and refer to the following code: Example IV. read custom field values C#

Example V. read not exported C#

Read all slips and add a filter to retrieve not yet exported records only.

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Slip";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "500";

OA.Attribute attrFilter = new OA.Attribute();
attrFilter.name = "filter";
attrFilter.value = "not-exported";

rr.attributes = new OA.Attribute[] { attrLimit, attrFilter };

// Tell the server we are filtering on import_export records created
by MY_APP
oaImportExport importExport = new oaImportExport();
importExport.application = "MY_APP";
rr.objects = new oaBase[] { importExport };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });

///Mark the slip with id = 4 as exported by the application MY_APP on
4/1/2011.
///After doing this, the next read call with not-exported filter
attribute will not export this record.
oaImportExport exportRecord = new oaImportExport();
exportRecord.application = "MY_APP";
exportRecord.type = "Slip";
exportRecord.id = "4";
exportRecord.exported = "2011-04-01 00:00:00";

UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] {
exportRecord });
```

Example VI. read not-exported Envelopes with date filter C#

Request envelope records that were approved in a certain date range and were not exported yet.

Note: Multiple filters can be used. They should be CSV concatenated in one single filter attribute. For example, to retrieve all timesheet entries in a certain date range for approved timesheets only, attrFilter.value should be "newer-than, older-than, approved- timesheets" .

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Envelope";

//Filter by date range and by the special not-exported flag
OA.Attribute attrFilter = new OA.Attribute();
attrFilter.name = "filter";
attrFilter.value = "newer-than,older-than,not-exported";

//Name of the field to apply date filter to
OA.Attribute attrField = new OA.Attribute();
attrField.name = "field";
attrField.value = "date_approved,date_approved";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "500";

rr.attributes = new OA.Attribute[] { attrFilter, attrField,
attrLimit};

// set newer-than filter date
oaDate dateNewer = new oaDate();
dateNewer.year = "2008";
dateNewer.month = "10";
dateNewer.day = "17";

// set older-than filter date
oaDate dateOlder = new oaDate();
dateOlder.year = "2008";
dateOlder.month = "10";
dateOlder.day = "17";

rr.objects = new oaBase[] { dateNewer, dateOlder };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example VII. read with lookup by custom field value C#

```
ReadRequest rr = new ReadRequest();
rr.method = "equal to";
rr.type = "Project";

//Get the first 100 records only
OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "100";
rr.attributes = new OA.Attribute[] { attrLimit };

//Get only records with custom field ProjectStatus set to "Yellow"
//Specify additional values in comma delimited list, e.g.
"Yellow,Green,Red"
//Provide empty string to get records that don't have any value set,
//e.g. filter.ProjectStatus__c = "";
//If the custom field type is Date provide default date to get records
that
//have no value, e.g. filter.ProjectStatus__c = "0000-00-00";
oaProject filter = new oaProject();
filter.ProjectStatus__c = "Yellow";
rr.objects = new oaBase[] { filter };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example VIII. read equal to with explicit OR condition C#

Search for all Bookings for users with id 5 or 6.

```
ReadRequest rr = new ReadRequest();
rr.method = "equal to, or equal to";
rr.type = "Booking";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "100";
rr.attributes = new OA.Attribute[] { attrLimit };

oaBooking book1 = new oaBooking();
book1.userid = "3";

oaBooking book2 = new oaBooking();
book2.userid = "10";

rr.objects = new oaBase[] { book1, book2 };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example IX. batch multiple read equal to requests C#

In many cases, the desired records cannot be retrieved in a single read request.

In such situations, we recommend batching multiple read requests in a single read call when integration logic allows it.

The code below results in a single trip to server, and therefore is clocked as one single transaction against your daily account limit.

```
int[] idsList = new int[] { 1, 22, 1633, 32, 9, 28, 39 };
ReadRequest[] readRequestsList = new ReadRequest[idsList.Length];

for (int i = 0; i < 1000 && i < idsList.Length; i++)
{
    ReadRequest rr = new ReadRequest();
    rr.type = "Slip";
    rr.method = "equal to";

    OA.Attribute attrLimit = new OA.Attribute();
    attrLimit.name = "limit";
    attrLimit.value = "1";
    rr.attributes = new OA.Attribute[] { attrLimit };

    oaSlip slipToRead = new oaSlip();
    slipToRead.id = idsList[i].ToString();
    rr.objects = new oaBase[] { slipToRead };

    readRequestsList[i] = rr;
}

ReadResult[] results = _svc.read(readRequestsList);
```

Java Read Code Examples

Note that "limit" attribute is **always required**, but for the sake of saving space, only the first example shows the loop which correctly gets multiple batches of data.

Example I. read equal to Java

```
// Create a read request for an envelope with internal id 211
ReadRequest[] reads = new ReadRequest[1];
reads[0] = new ReadRequest();
reads[0].setType("Envelope"); // we are requesting Envelope type
reads[0].setMethod("equal to"); // method to return a specific
envelope
OaEnvelope env = new OaEnvelope();
env.setId("211");
reads[0].setObjects(new OaBase[]{env});
```

```
int limit = 1000; // only read 1000 records at a time
int index = 0;// record index to start the read from
// add an attribute to our read, specifying the base record # (index)
// and the max # of records to be returned (limit)
Attribute attr = new Attribute();
attr.setName("limit");
attr.setValue(String.format("%1$d", limit));
reads[0].setAttributes(new Attribute[]{attr});

// perform the read
System.out.print("Fetching envelopes...");
ReadResult[] results = binding.read(reads);

// output the results
while(true)
{
     int numRead = 0;

     for (int i = 0; i < results.length; ++i)
     {
          ReadResult r = results[i];
          if (r.getObjects() != null)
          {
                  System.out.println("Read " + r.getObjects().length
                  + " envelopes\n");
                  OaBase[] objs = r.getObjects();
                  for (numRead = 0; numRead < objs.length; ++numRead)
                  {
                      OaEnvelope envelope = (OaEnvelope)objs[numRead];
                      System.out.println("Envelope name: " +
                      envelope.getName());
                      System.out.println("Envelope number: " +
                      envelope.getNumber());
                      System.out.println("Envelope total: " +
                      envelope.getTotal());
                      System.out.println();
                      // ..etc.
                      index++;
                  }
                  System.out.println("Read "+numRead+" envelopes");
          }
          else
          {
                  System.out.println("Read 0 envelopes\n");
          }
```

```
        }
        // if we've read up to the limit, do another read using index as
        our base
        if( numRead == limit )
        {
             System.out.println("Fetching "+ limit +" more envelopes");
             attr.setValue(String.format("%1$d, %2$d", index, limit));
             results = binding.read(reads);
        }
        else
        {
             // no more to read
             break;
        }
}
```

Example II. read not equal to Java

Note that the "limit" attribute is required as illustrated in Example I. read equal to Java.

```
// Create a read request for envelopes with a non-zero total.
ReadRequest[] reads = new ReadRequest[1];
reads[0] = new ReadRequest();
reads[0].setType("Envelope"); // we are requesting Envelope type
reads[0].setMethod("not equal to"); // method to return a subset of
data based on search criteria.

// We are searching for an envelope with total not equal to 0.
OaEnvelope obj = new OaEnvelope();
obj.setTotal("0.00");
reads[0].setObjects(new OaEnvelope[] { obj });

// perform the read
System.out.print("Fetching envelopes...");
ReadResult[] results = binding.read(reads);

// output the results
for (int i = 0; i < results.length; ++i)
{
     ReadResult r = results[i];
     if (r.getObjects() != null)
     {
          System.out.println("Read " + r.getObjects().length + "
          envelopes\n");
          OaBase[] objs = r.getObjects();
```

```
            for (int j = 0; j < objs.length; ++j)
            {
                 OaEnvelope env = (OaEnvelope)objs[j];
                 System.out.println("Envelope name: " + nv.getName());
                 System.out.println("Envelope number: " + env.getNumber());
                 System.out.println("Envelope total: " + nv.getTotal());
                     System.out.println();
                 // ..etc.
                 }
      }
      else
      {
            System.out.println("Read 0 envelopes\n");
      }
}
```

Example III. read not exported Java

Note that the "limit" attribute is required as illustrated in Example I. read equal to Java.

```
// Read all slips, but add a filter so we only retrieve not-yet-
exported records.
// Below is an example on how to mark items as being exported.
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "not-exported" );
ReadRequest read = new ReadRequest();
read.setMethod( "all" );
read.setType( "Slip" );
read.setAttributes( new Attribute[]{attr} );

// Tell the server we're filtering on import_export records created by
MY_APP
OaImportExport importExport = new OaImportExport();
importExport.setApplication( "MY_APP" );
read.setObjects( new OaBase[] { importExport } );

ReadResult[] results = binding.read(new ReadRequest[] { read });

// To modify and read not-yet exported slips
// Mark the slip with id = 4 as exported by the application
// MY_APP on 4/1/2011. By doing this, we can add a filter attribute to
// our read call that will filter out records exported by MY_APP.
OaImportExport exportRecord = new OaImportExport();
exportRecord.setApplication( "MY_APP" );
exportRecord.setType( "Slip" );
```

```
exportRecord.setId( "4" );
exportRecord.setExported( "2011-04-01 00:00:00" );
UpdateResult[] ur = binding.upsert( new Attribute[]{}, new OaBase[] {

exportRecord });
```

Example IV. read date filter Java

Note that the "limit" attribute is required as illustrated in Example I. read equal to Java.

```
// Request envelope records updated based on a certain date.
ReadRequest read = new ReadRequest();
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "newer-than,older-than" ); // filter for records in
date range separated by a comma
Attribute field = new Attribute();
field.setName( "field" );
field.setValue( = "updated,updated" ); //one for each date object
separated by a comma
read.setMethod( "all" );
read.setType( "Envelope" );
read.setAttributes( new Attribute[]{attr});

OaDate dateNewer = new OaDate();
OaDate dateOlder = new OaDate();

// set newer than date.
dateNewer.setYear( "2008" );
dateNewer.setMonth( "10" );
dateNewer.setDay( "16" );

// set older than date.
dateOlder.setYear( "2008" );
dateOlder.setMonth( "10" );
dateOlder.setDay( "17" );

read.setObjects(new OaBase[] { dateNewer,dateOlder });
ReadResult[] results = stub.read(new ReadRequest[] { read });
```

add

Syntax

```
UpdateResult[] addResults = stub.add(objects);
```

Use

Use this call to add data to OpenAir. The method returns an error if more than 1000 objects are passed in.

You can use an externalid field as a foreign key and add a record without querying first for an internal id. See Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

Note: Use createUser method to add OpenAir users (oaUser object).

Arguments

Name	Type	Description
Objects	[] oaBase	Array of oaBase objects

Note: Refer to Adding Records with Inline Custom Field Values for procedures on adding records with inline custom field values.

Response

```
Array of UpdateResult objects
```

Sample Code - C#

```
//Define a category object to create in OpenAir
oaCategory category = new oaCategory();
category.name = "New Category";
category.cost_centerid = "123";
category.currency = "USD";

//Invoke the add call
UpdateResult[] results = _svc.add(new oaBase[] { category });

//Get the new ID
string newID = results[0].id;
```

Sample Code - Java

```
// Create a category object to send to the service
OaCategory category = new OaCategory();

// Set several properties
category.setName("my new category");

// Add the category to an array of oaBase objects
OaBase[] records = new OaBase[] { category };

// Invoke the add call
UpdateResult[] results = stub.add(records);

// Get the new ID
String newID = results[0].getId();
```

upsert

Syntax

```
UpdateResult[] upsertResults = stub.upsert(attributes, objects);
```

Use

Use this call to add or modify data to OpenAir based on lookup attributes. The method returns an error if more than 1000 objects are passed in.

You can use an externalid field as a foreign key and add a record without querying first for an internal id. See Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

Arguments

Name	Type	Description
attributes	[] LoginParams	Array of Attribute objects
objects	[]oaBase	Array of oaBase objects

Note: Refer to Adding Records with Inline Custom Field Values for procedures on adding records with inline custom field values.

Response

`Array of UpdateResult objects`

Types of Attributes Used

Name: lookup
Value: name of the field which should be used for lookup.

- If you pass external_id, upsert will attempt to find another record with external_id as specified in the object being upserted.
- If it finds another record, it will do a modify, otherwise the record will be added.

Sample Code - C#

```
//Define a category object to create/update in OpenAir
oaCategory category = new oaCategory();
category.name = "Updated Category";
category.externalid = "555";

// Specify that the lookup is done by external_id and not by (default)
internal id
OA.Attribute attrLookup = new OA.Attribute();
attrLookup.name = "lookup";
attrLookup.value = "externalid";

// Invoke the upsert call, passing and saving the results in a
UpdateResult object
UpdateResult[] results = _svc.upsert(new OA.Attribute[] { attrLookup
}, new oaBase[] { category });
```

Sample Code - Java

```
// upsert call
// Create a category object to send to the service
OaCategory category = new OaCategory();

// Set several properties
category.setName("SOAP created category 12/4");
category.setExternalid("555");

// Specify lookup attribute
Attribute lookup = new Attribute();
lookup.setName("lookup");
lookup.setValue("externalid");

// Add the account to an array of oaBase objects
OaBase[] records = new OaBase[] { category };

// Invoke the upsert call, passing.
// and saving the results in a UpdateResult object
```

```
UpdateResult[] results = stub.upsert(new Attribute[] { lookup },
records);
```

createAccount

Syntax

```
UpdateResult result = stub.createAccount(user, company);
```

Use

Use this call to create OpenAir accounts. This method also creates the first administrative user. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
user	oaUser	oaUser object
company	oaCompany	oaCompany object

Response

`UpdateResult` object.

Sample Code - C#

```
// Create a new OpenAir account
oaCompany comp = new oaCompany();
comp.nickname = "New Account";

oaUser user = new oaUser();
user.nickname = "Admin";
user.role_id = "1";
user.password = "%^&*&^";
user.addr_email = "sss@sss.com";
user.account_workscheduleid = "1";

UpdateResult result = _svc.createAccount(user, comp);
```

Sample Code - Java

```
OaCompany comp = new OaCompany();
OaUser cuser = new OaUser();
comp.setNickname("New Account");

cuser.setNickname("Admin");
cuser.setRole_id("1");
cuser.setPassword("%^&*&^");
```

```
cuser.setAddr_email("sss@sss.com");
cuser.setAccount_workscheduleid("1");

UpdateResult result = stub.createAccount(cuser, comp);
```

createUser

Syntax

```
UpdateResult result = stub.createUser(user, company);
```

Use

Use this call to create OpenAir users. The method returns an error if more than 1000 objects are passed in. For procedures in setting a User workschedule, refer to oaUser.

You can use an externalid field as a foreign key and add a record without querying first for an internal id. See Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

Arguments

Name	Type	Description
user	oaUser	oaUser object
company	oaCompany	oaCompany object

Response

`UpdateResult` object.

Sample Code - C#

```
oaCompany comp = new oaCompany();
comp.nickname = "New Account";// specify nickname of the account the
user is being added to.

oaUser newUser = new oaUser();
newUser.nickname = "userA";
newUser.role_id = "1";   // role of administrator.
newUser.password = "******";
newUser.addr_email = "sss@sss.com";
newUser.account_workscheduleid = "1"; // Associate a valid
workschedule for the user.

UpdateResult result = _svc.createUser(newUser, comp);
```

Sample Code - Java

```
OaCompany comp = new OaCompany();
OaUser cuser = new OaUser();
comp.setNickname("openair");      // specify nickname of the account
the user is being added to.
cuser.setNickname("userA");
cuser.setRole_id("1");   // role of administrator.
cuser.setPassword("******");
cuser.setAddr_email("sss@sss.com");
cuser.setAccount_workscheduleid("1");  // Associate a valid
workschedule for the user.

UpdateResult result = stub.createUser(cuser, comp);
```

submit

Syntax

```
SubmitResult[] results = stub.submit(requests);
```

Use

Use this call to submit OpenAir entities for approval. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
requests	[] SubmitRequest	Array of SubmitRequest objects

Response

Array of SubmitResult

Sample Code - C#

```
// submit an envelope for approval
oaEnvelope env = new oaEnvelope();
env.id = "122";

oaApproval appr = new oaApproval();
appr.cc = "help@ddd.com"; // cc approval email to additional contacts
appr.notes = "Approval notes";

SubmitRequest sr = new SubmitRequest();
sr.submit = env;
```

```
sr.approval = appr;

SubmitResult[] results = _svc.submit(new SubmitRequest[] { sr });
```

Sample Code - Java

```
// submit an envelope for approval
OaEnvelope env = new OaEnvelope();
env.setId("122");
OaApproval appr = new OaApproval();
appr.setCc("help@ddd.com"); // cc approval email to additional
contacts
appr.setNotes("approval notes");
SubmitRequest sub = new SubmitRequest();
sub.setApproval(appr);
sub.setSubmit( env );

SubmitResult[] results = stub.submit(new SubmitRequest[] { sub });
```

makeURL

Syntax

```
MakeURLResult[] mkresults = stub.makeURL(request);
```

Use

Use this call to create a valid URL to a specified OpenAir page. It requires a valid user login to succeed.

Arguments

Name	Type	Description
request	[] MakeURLRequest	Array of MakeURLRequest object

Currently, the list of valid page strings, with associated applications and arguments, includes:

- **default-url**
  app= km, ma, pb, rm, pm, ta, te, or tb (Points to the starting page in any one of the applications, which is the page you see when you click on the application link.)

  For example: If you are the administrator and are using pm as the app attribute, the first page is the Projects list in the Projects application. For non-administrative users, it would be the list of tasks to which the user is assigned.)

- **company-settings**
  app= ma (points to Administration > Global Settings)
- **currency-rates**
  app= ma (points to Administration > Global Settings> Currencies)
- **import-export**
  app= ma (points to Administration > Global Settings > Integration: Import/Export)
- **custom-fields**
  app= ma (points to Administration > Global Settings > Custom Fields)
- **list-reports**
  app= ma (points to Reports > last page accessed)
- **list-customers**
  app= ma (points to Administration > Global Settings > Customers)
- **list-projects**
  app= pm (points to Projects > Projects)
- **list-prospects**
  app= om (points to Opportunities > Prospects)
- **list-resources**
  app= rm (points to Resources > Resources)
- **list-timesheets**
  app= ta (points to Timesheets > Timesheets > Open)
- **create-timesheet**
  app= ta (points to Timesheets > Create Timesheet)
- **list-timebills**
  app= tb (points to Invoices > Charges)
- **list-invoices**
  app= tb (points to Invoices > Invoices)
- **create-invoice**
  app= tb (points to Invoices > Invoices > Create Invoice)
- **list-envelope-receipts**
  app= te (points to Expenses > Envelopes > Receipts)
  `arg = oaEnvelope envelope = new oaEnvelope(); envelope.id = <envelope internal id>`
- **list-envelopes**
  app= te (points to Expenses > Expense Reports > Open)
- **create-envelope**
  app= te (points to Expenses > Expense Reports > Create Envelope)
- **create-envelope-receipt**
  app= te (points to Expenses > Expense Reports > Create Receipt

- **dashboard**
  app= ma (points to Dashboard)
- **list-purchase-requests**
  app= po (points to Purchases> Purchase Requests)
- **quick-search-resources**
  app= rm (points to Resources > Quick Search)
- **custom-search-resources**
  app= rm (points to Resources > Custom Search)
- **view-invoice**
  app= tb (displays the invoice with specified internal id)
  `arg= oaInvoice invoice = new oaInvoice(); invoice.id = <invoice internal id>`
- **dashboard-project**
  app= pm (displays the dashboard view of the project with specified internal id)
  `arg= oaProject project = new oaProject(); project.id = <project internal id>`
- **grid-timesheet**
  app= ta (displays the grid of the timesheet with specified internal id)
  `arg= oaTimesheet timesheet = new oaTimesheet(); timesheet.id = <timesheet internal id>`
- **report-timesheet**
  app= ta (displays the timesheet report of specified internal id)
  `arg= oaTimesheet timesheet = new oaTimesheet(); timesheet.id = <timesheet internal id>`

Response

`Array of MakeURLResult`

Sample Code - C#

```
oaEnvelope envelope = new oaEnvelope();
envelope.id = "1";

MakeURLRequest mur = new MakeURLRequest();
mur.uid = _svc.SessionHeaderValue.sessionId;
mur.app = "te";
mur.page = "list-envelope-receipts";
mur.arg = envelope;

MakeURLResult[] results = _svc.makeURL(new MakeURLRequest[] { mur });
```

Sample Code - Java

```
String sessionId = loginResult.getSessionId();
MakeURLRequest make = new MakeURLRequest();

make.setUid(sessionId);
make.setApp("te");
make.setPage( "list-envelope-receipts" );
OaEnvelope envelope = new OaEnvelope();
envelope.setId("1");
make.setArg( envelope );

// make url
MakeURLResult[] mkresults = stub.makeURL(new MakeURLRequest[] { make });
```

modify

Syntax

```
UpdateResult[] result = stub.modify(attributes, objects);
```

Use

Use this call to modify data in OpenAir. The method returns an error if more than 1000 objects You can use an externalid field as a foreign key and add a record without querying first for an internal id. Refer to instructions for Using an externalid field as a foreign key. See the following code examples: Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java. You can also modify data in OpenAir based on an internal ID.

Arguments

attributes	[] Attribute	Array of Attribute objects. See note below.
objects	[] oaBase	Array of oaBase objects

Note: Refer to Modifying Records to Set Custom Field Values for information on modifying the wsdl file to include custom fields and using the lookup_custom attribute. Specific code examples follow.

Response

Array of UpdateResult objects.

C# Modify Code Examples

Example I. modify C#

Modify a customer's email address.

```
oaCustomer customer = new oaCustomer();
customer.id = "37";
customer.addr_email = "newest@email.com";

UpdateResult[] res =_svc.modify(new OA.Attribute[]{},new
oaBase[]{customer});
```

Example II. modify using external_id as foreign key lookup field C#

This modify request updates the filterset_ids property of user(id = 12). The API looks up the internal ids of Filtersets which have external_ids "extrn1", "extrn2" and "extrn3" and assigns the filterset_ids property of the target user with the list of corresponding internal ids, like "12,3,24".

```
oaFieldAttribute lookupAttr = new oaFieldAttribute();
lookupAttr.name = "external";
lookupAttr.value = "filterset_ids:Filterset:1";

oaUser user = new oaUser();
user.id = "12";
user.filterset_ids = "extrn1,extrn2,extrn3";
user.attributes = new oaBase[] { lookupAttr };

UpdateResult[] results = _svc.modify(new OA.Attribute[] {}, new
oaBase[] { user });
```

Example III. modify using custom field as lookup field C#

To use custom field as a lookup field in place of internal id, use the following syntax. The API will find the customer(s) which have CustField12 value set to "somevalue" and update the name on matching records to "John Carr".

```
oaCustomer customer = new oaCustomer();
customer.name = "John Carr";
customer.CustField12__c = "somevalue";

//this attribute specifies which custom field should be used for
lookup
OA.Attribute lookupAttr = new OA.Attribute();
lookupAttr.name = "lookup_custom";
lookupAttr.value = "CustField12__c";

UpdateResult[] results = _svc.modify(new OA.Attribute[] { lookupAttr
}, new oaBase[] { customer });
```

Example IV. update custom field value using an inline (__c) property C#

```
oaProject project = new oaProject();
project.id = "123"; //id of record to modify
project.ProjectStatus__c = "Yellow"; //new custom field value

//Define attribute that directs API to update a custom field.
OA.Attribute updateCustom = new OA.Attribute();
updateCustom.name = "update_custom";
updateCustom.value = "1";

UpdateResult[] res = _svc.modify(new OA.Attribute[] { updateCustom },
new oaBase[] { project });
```

Example V. update custom field value using custom equal to method C#

```
OA.Attribute lookupAttr = new OA.Attribute();
lookupAttr.name = "method";
lookupAttr.value = "custom equal to";

oaCustomField customField = new oaCustomField();
customField.type = "User"; //name of the object the field is
associated with
customField.id = "12";    //internal id of the user record.
customField.name = "cust_field_name"; // internal name of the custom
field.
customField.value = "My new value"; // new value

UpdateResult[] updateResults = _svc.modify(new OA.Attribute[] {
lookupAttr }, new oaBase[] { customField });
```

Example VI. modify import_export and read not-exported C#

Mark the envelope with id = 4 as exported by the application MY_APP on 4/1/2008. By doing this, we can later use "not-exported" filter to read only records that have not yet been exported by MY_APP.

```
oaImportExport exportRecord = new oaImportExport();
exportRecord.application = "MY_APP";
exportRecord.type = "Envelope";
exportRecord.id = "4";
exportRecord.exported = "2008-04-01 00:00:00";
UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] {
exportRecord });
```

```
//Define read parameters
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Envelope";

//Export only records that have not yet been exported by MY_APP
OA.Attribute notExportedAttr = new OA.Attribute();
notExportedAttr.name = "filter";
notExportedAttr.value = "not-exported";
rr.attributes = new OA.Attribute[] { notExportedAttr };

//Direct the API to filter out records exported by MY_APP
oaImportExport importExport = new oaImportExport();
importExport.application = "MY_APP";
rr.objects = new oaBase[] { importExport };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Java Modify Code Examples

Example I. modify Java

```
// Modify customer's email address
OaCustomer customer1 = new OaCustomer();
customer1.setAddr_email("new@email.com");
customer1.setId("66");

// Attribute not used in this case but needs to be passed in.
Attribute dummy = new Attribute();
UpdateResult[] results = binding.modify(new Attribute[] { dummy },
new OaBase[] { customer1 });
```

Example II. externalid as foreign key lookup field Java

```
// For each xxxid field that needs to be looked up, set the id field
(filtersetids) to the externalid field value.
// Create an oaFieldAttribute object and set its members. In this
example, filtersetids accepts a list of OpenAir internal ids
separated by a comma. In most cases just a single id values is used.
OaFieldAttribute attr = new OaFieldAttribute();
att.setName("external"); // type of lookup (external will lookup the
field using external_id field)
attr.setValue("filtersetids:Filterset:1"); // colon separated values:
field name (as it exists in the object, matching record type to
process lookup for, 1 is needed to process comma separated values
(it's not required for regular fields)
```

```
// Set the user field (filtersetids in this case, to the value of the
externalid (instead of the internalid used normally)
OaUser user = new OaUser();
user.setId("10119");
user.setFiltersetids("external1,external2,external3,external4");

// notice that the field used (filtersetids) matches the first part of
the attributes value.
// Set the attributes collection for user object. Set the attribute as
part of collection.
user.setAttributes(new OaBase[] {attr});

// process modify
UpdateResult[] results = service.modify(new Attribute[] { null},
new OaBase[] {user});
```

Example III. custom equal to Java

```
// Attributes can be used to specify non-default method to update
custom fields for a given object:
// create the attribute to specify method.
Attribute custom = new Attribute();
custom.setName("method");
custom.setValue("custom equal to");

// create custom field object
OaCustomField customf = new OaCustomField();
customf.setType("User");   // name of the object custom field is
associated with. In this example, it is User. See table of custom
equal to objects.

customf.setName("userc"); // internal name of the custom field.
customf.setId("1");       // internal id of the user record
customf.setValue("My new value");   // custom value

UpdateResult[] updateResults = stub.modify(new Attribute[] { custom }, new
OaBase[] { customf });
```

Example IV. not exported Java

```
// To modify and read not-yet exported envelopes
// mark the envelope with id = 4 as exported by the application
// MY_APP on 4/1/2008. By doing this, we can add a filter attribute to
// our read call that will filter out records exported by MY_APP.
OaImportExport exportRecord = new OaImportExport();
exportRecord.setApplication( "MY_APP" );
```

```
exportRecord.setType( "Envelope" );
exportRecord.setId( "4" );
exportRecord.setExported( "2008-04-01 00:00:00" );
UpdateResult[] ur = binding.upsert( new Attribute[]{}, new OaBase[] {

exportRecord });

// Now do a read of all envelopes, but add a filter
// so we only retrieve not-yet-exported records
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "not-exported" );

ReadRequest read = new ReadRequest();
read.setMethod( "all" );
read.setType( "Envelope" );
read.setAttributes( new Attribute[]{attr} );

// tell the server we're filtering on import_export records created by
MY_APP
OaImportExport importExport = new OaImportExport();
importExport.setApplication( "MY_APP" );
read.setObjects( new oaBase[] { importExport } );

ReadResult[] results = binding.read(new ReadRequest[] { read });
```

delete

Syntax

```
UpdateResult[] deleteResults = stub.delete(object);
```

Use

Use this call to delete data in OpenAir based on an internal ID. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
objects	[] oaBase	Array of oaBase objects

Response

Array of UpdateResult objects.

Sample Code - C#

```
// delete customer with internal id 66.
oaCustomer customer = new oaCustomer();
customer.id = "66";

UpdateResult[] deleteResults = stub.delete(new oaBase[1] { customer });
```

Sample Code - Java

```
// delete customer with internal id 66.
OaCustomer customer = new OaCustomer();
customer.setId("66");

UpdateResult[] deleteResults = stub.delete(new OaBase[] { customer });
```

whoami

Syntax

```
oaUser user = stub.whoami();
```

Use

Use this call to get the currently logged in OpenAir user.

Arguments

There are no arguments.

Response

oaUser

Sample Code - C#

```
oaUser user = stub.whoami();
```

Sample Code - Java

```
OaUser user = stub.whoami();
```

servertime

Syntax

```
oaDate dateNow = stub.servertime();
```

Use

Use this call to get the current server time oaDate object.

Arguments

There are no arguments.

Response

`oaDate`

Sample Code - C#

```
oaDate dateNow = stub.servertime();
```

Sample Code - Java

```
OaDate dateNow = stub.servertime();
```

logout

Syntax

```
stub.logout();
```

Use

Use this call to log out of a session.

Arguments

There are no arguments.

Response

You are logged out.

Sample Code - C#

```
// This invalidates sessionID and user needs to login to make any
calls to the API again.
stub.logout();
```

Sample Code - Java

```
// This invalidates sessionID and user needs to login to make any
calls to the API again.
stub.logout();
```

Chapter 4 Web Services Method Complex Types

The following are Web Services parameters and return values. Each is presented with a statement about its use and a listing of children. Links to calls where these types are used are also provided.

Attribute

Use this complex type to specify attribute for various calls. Attribute has the following children.

Field Name	Description
name	Name of the method.
value	Value of an attribute.

For more information on its use, refer to the following methods: upsert, submit, modify, and makeURL.

LoginParams

Use login parameters to authenticate users into the system. LoginParams has the following children.

Field Name	Description
api_namespace	Namespace name.
api_key	Specify an API key assigned to you.
company	Specify companyID.
user	Specify User ID.
password	Specify password.
client	Specify the client name.
version	Specify the version of the client.

For more information on its use, refer to the login method.

LoginResult

This complex type holds the results of login call. LoginResult has the following children.

Field Name	Description
sessionId	ID associated with this session.
URL	URL of the logged in session for this user.

For more information on its use, refer to the login method.

MakeURLRequest

Use this complex type to specify parameters for makeURL call. MakeURLRequest has the following children.

Field Name	Description
uid	Valid sessonID.
page	Page abbreviation.
app	Application abbreviation.
arg	Any argument.

For more information on its use, refer to the makeURL method.

MakeURLResult

This complex type holds the results of makeURL call. MakeURLResult has the following children.

Field Name	Description
url	Valid authenticated URL.
errors	A collection of oaError objects.
status	-l in case of any errors.

For more information on its use, refer to the makeURL method.

ReadRequest

Use this complex type to specify parameters for read method. ReadRequest has the following children.

Field Name	Description
method	Method name.
fields	Comma separated list of fields to return.
attributes	A collection of Attribute objects.
type	Types of the record.
objects	Any oaBase objects as arguments.

For more information on its use, refer to the read method

Using ReadRequest

ReadRequest can be used with a number of oaComplex Types as well as with different methods. You can also specify fields and attributes. All of these help you specify what records you want or allow you to limit the ReadResults that are returned. Refer to the following sections for specific information.

Types

The following types can be specified. To review the field names and definitions of each complex type, refer to Chapter 6 "OpenAir Complex Types". A list of supported methods is provided for each complex type.

Actualcost	Address	Agreement_to_project
Agreement	Approval	Attachment
Attributeset	Attribute	BookingType
Booking	BudgetAllocation	Budget
Category_1	Category_2	Category_3
Category_4	Category_5	Category
Ccrate	Company	Contact
Costcategory	Costcenter	Costtype
Currencyrate	Currency	CustField
Customerpo_to_project	Customerpo	Customer
CustomField	Dealcontact	Dealschedule
Deal	Department	Entitytag
Envelope	Error	Estimateadjustment
Estimateexpense	Estimatelabor	Estimatemarkup
Estimatephase	Estimate	Event

Filterset	ForexInput	FormPermissionField
Fulfillment	HierarchyNode	Hierarchy
History	ImportExport	Invoice
IssueCategory	IssueSeverity	IssueSource
IssueStage	IssueStatus	Issue
Item	Jobcode	Leave_accrual_rule_to_user
Leave_accrual_rule	Leave_accrual_transaction	LoadedCost
Paymentterms	Paymenttype	Payment
Payrolltype	Preference	Product
Projectassign	Projectbillingrule	Projectbillingtransaction
Projectgroup	Projectlocation	Projectstage
Projecttaskassign	Projecttask_type	Projecttask
Project	Proposalblock	Proposal
Purchase_item	Purchaseorder	Purchaserequest
Purchaser	RateCardItem	Ratecard
Reimbursement	Repeat	Report
Request_item	Resourceprofile_type	Resourceprofile
RevenueContainer	Revenue_recognition_rule_amount	Revenue_recognition_rule
Revenue_recognition _transaction	RevenueStage	Schedulebyday
Scheduleexception	Schedulerequest_item	Schedulerequest
SlipProjection	Slipstage	Slip
TagGroupAttribute	TagGroup	TargetUtilization
TaskTimecard	Task	TaxLocation
TaxRate	Ticket	Timecard
Timesheet	Timetype	Todo
Uprate	UserWorkschedule	User
Vendor	Viewfilterrule	Viewfilter
Workspacelink	Workspaceuser	

Methods

Use one of the following methods. Each is explained as follows:

all

- Returns all available records.
- Use this cautiously as too many records may be requested for the server or client to handle.

equal to

- Returns records that have fields that are equal to the field value(s) passed in.
- Use ReadRequest objects to specify object and field values to include in this search.

- Example I. read equal to C# and Example I. read equal to Java.
- Multiple equal to and not equal to method arguments can be mixed in a single ReadRequest, which allows creating queries with AND/OR filtering logic. See Example VIII. read equal to with explicit OR condition C#. To use this feature:
  - Modify the method parameter to include multiple “equal to” and “not equal to” methods as desired, separated by commas. For example: “equal to, not equal to”.
  - Next, supply an equal number of argument objects to filter on.
  - You may precede each method by an “and” or an “or” operator. For example: “equal to, or equal to”. If no operator is supplied, a logical AND relationship is assumed.

    Note: This feature supports only one level of logical operators. It does not support nesting of operators like "equal to and (equal to or equal to)". In addition, if one of the filtering objects has multiple properties specified, the top level logical operator will be applied to all of them.

not equal to

- Returns records that have fields that are not equal to the field value(s) passed in.
- Use ReadRequest objects to specify object and field values to include in this search.
- Example II. read not equal to C# and Example II. read not equal to Java.

custom equal to

- Allows you to display custom field values for a particular record.
- read.objects collection must contain a record with the internal id set to specify the id of the record the custom fields should be returned for.
- oaCustomField objects are returned as part of the ReadResult.objects.collection.
- Example III. read custom field definitions C#.
- **custom equal to objects** - For a list of associated objects, see the oaCustField Association Table or Type of Object.

  Note: Custom fields can also be read inline with the parent object. See Modifying Records to Set Custom Field Values.

Fields

Use a comma separated list of fields to limit the amount of data returned.

Attributes

Use one of the following attributes.

Attribute Name	Value	Result
limit	'1000' or '0, 1000'	Restricts the number of records returned. Single number value: "1", "500", "1000" - simply restricts the number of records returned. Double number value: "0, 1000" - the first integer specifies the offset of the first record to return and the second integer limits the number of records to return. To request data in consecutive batches, only the first part of the limit attribute should be incremented - "0,1000", "1000,1000", "2000,1000", etc. Sequence requests should be submitted until the result comes back empty or has less items than 1000.
deleted	1	Returns deleted records. It can be used together with newer-than filter.
include_flags	1	Returns account or user switches, by default those are not populated.
include_nondeleted	1	Returns all records, deleted and nondeleted. **Note:** This attribute only works in conjunction with the "deleted" attribute.
with_project_only	1	Used only with type: Customer. Will only return customers which have associated project records.
base_currency	3	Letter currency code. Works with type: Currencyrate. Converts values on the fly to currency specified.
generic	1	Returns generic resources (users) only, where by default, the API returns regular users only.
filter See Notes below.	* open-envelopes	Returns only records associated with an open envelope.
	* approved-envelopes	Returns only records associated with an approved envelope.
	* rejected-envelopes	Returns only records associated with a rejected envelope.
	* submitted-envelopes	Returns only records associated with a submitted envelope.
	* nonreimbursed-envelopes	Returns envelopes that have a non-zero balance attribute.
	* reimbursable-envelope	Returns only records associated with a reimbursable envelope.
	* open-slips	Returns only records associated with an open slip.
	* approved-slips	Returns only records associated with an approved slip.
	* open-timesheets	Returns only records associated with an open timesheet.
	* approved-timesheets	Returns only records associated with an approved timesheet.
	* rejected-timesheets	Returns only records associated with a rejected timesheet.

Attribute Name	Value	Result
	* submitted-timesheets	Returns only records associated with a submitted timesheet.
	* not-exported	Returns only records that have not been marked as exported. See Example V. read not exported C# and Example III. read not exported Java.
	* approved-revenue-recognition-transactions	Returns only revenue recognition transactions belonging to approved revenue_container records.
	date filters: * newer-than * older-than * date-equal-to * date-not-equal-to	Returns only records that have a value in the 'updated' field that is newer-than, older-than, date-equal-to, or date-not-equal-to the date specified in the oaDate object in the objects collection. To compare date fields other than 'updated', add the following additional attribute information: name="field" value="[some date field]" See Example VI. read not-exported Envelopes with date filter C# and Example IV. read date filter Java.

Note: The following notes apply to using one or more filters:

- A record is associated with an open envelope, open slip, or open timesheet if it has an id field that points to either an envelope (envelope_id), slip (slip_id), or timesheet (timesheet_id). Do not use the filter attribute with data types that do not have associated ids. (i.e., Do not use type Project and the filter open-envelopes.)
- Multiple date filters can be used. They should be separated by a comma: newer-than, older-than, date-equal-to, date-not-equal-to. The argument objects should be included in the same order as the filters those arguments apply to as part of the objects collection.
- Multiple filters can be used. They should be CSV concatenated in one single filter attribute. For example, you can use the following attributes to retrieve all timesheet entries in a certain date range for approved timehseets only: filter="newer-than, older-than, approved-timesheets".

ReadResult

This complex type returns any results of the read method. ReadResult has the following children.

Field Name	Description
objects	A collection of oaBase objects.
errors	A collection of oaError objects.

For more information on its use, refer to the read method.

SessionHeader

Use this complex type to hold session information. SessionHeader has the following children.

Field Name	Description
sessionId	Valid sessionID of the logged in user session.

For more information on its use, refer to the following methods: read, add, modify, upsert, createUser, createAccount, makeURL, whoami, logout, and delete.

SubmitRequest

Use this complex type to specify parameters for the submit method. SubmitRequest has the following children.

Field Name	Description
attributes	A collection of Attribute objects
submit	oaEnvelope or oaTimesheet object.
approval	oaApproval object.

For more information on its use, refer to the submit method.

SubmitResult

This complex type returns any results of the submit method. SubmitResult has the following children.

Field Name	Description
id	Internal id of the object submitted.
approval_warnings	String representing any warnings.
approval_errors	String representing any errors.
log	String representing the log of actions.
errors	A collection of oaError objects.
status	-1 in case of errors.

For more information on its use, refer to the submit method.

UpdateResult

Use this complex type holds the results of a given method call. UpdateResult has the following children.

Field Name	Description
id	Internal id of the record created or updated.
errors	A collection of oaError objects.
status	U - record was updated. A - record was added. D - record was deleted. -1 - one or more errors occurred.

For more information on its use, refer to the following methods: add, modify, upsert, createUser, createAccount, and delete.

VersionResult

This complex type holds the results of the version method call. VersionResult has the following children.

Field Name	Description
number	Current version number.
url	URL of the current version of the client installation.
size	Size of the file to download.

For more information on its use, refer to the version method.

Chapter 5 Custom Fields

Introduction to Custom fields

Custom Fields are helpful additions to your OpenAir account. You can use the oaCustomField complex type to modify or read custom field values associated with your records. You can use oaCustField to get a list of custom field definitions and their metadata in an OpenAir account such as name, association, the type of custom field (use with the read method). You can also use oaCustField to set valuelist on custom field definitions of drop-down and multi-select types (use with modify method).

You may also request all available custom field types that exist for a given object or you may read custom field values for a specific record. You can modify records to set custom field values and add records with inline custom field values.

Refer to the following sections for more information on working with custom fields. Links to code examples are provided for read and modify methods. Navigation for finding custom fields and XML tag limitations follow.

Finding Custom Fields

To find custom fields in the OpenAir Web application, go to Account > Company > Custom Fields or Administration > Global Settings > Custom fields, depending on the terminology in your user interface.

XML Tag Limitations

General XML tag limitations apply to the names of custom fields when the wsdl is modified. They include the following: names cannot start with a number or with the letters “xml” in any form such as XML or Xml. For example, 1MyCustomField or xmlMyCustomField would not work with SOAP.

Requesting Custom Fields for an Object

You can request all custom fields that exist for a given object. Use the read method and specify the CustField type and filter for a particular association. Refer to the oaCustField complex type Association Table or Type of Object for a list of possible associations.

Reading Custom Field Values

You can read custom field values for a given record using several options.

1. Use the custom equal to method described inUsing ReadRequest to only request custom field values for a particular record. You need to know the internal ID of the record in question. See Example IV. read custom field values C#.
2. Alternatively, modify the wsdl file to include custom fields for the object you're requesting. Use the field name and add "__c" to the end of the name. (Note that there are two underscores before the c.) See XML Tag Limitations.

Modifying Records to Set Custom Field Values

You can modify records to set custom field values. Refer to the following examples.

1. Use the custom equal to attribute of the modify call to set custom fields. See Example V. update custom field value using custom equal to method C# and Example III. custom equal to Java.
2. Use a custom field in place of an internal id to lookup and modify a record. Modify the wsdl file to include custom fields for the object you want to look up. Use the field name, add "__c" to the end of its name, and use the lookup_custom attribute. (Note that there are two underscores before the c.) Refer to Example lll. modify using custom field as lookup field C#. See XML Tag Limitations.

   Note: This method only works with modify and is not supported by the upsert method.
3. Modify custom fields inline in a single modify request with other fields. To utilize this feature, use the "update_custom" attribute in your modify request and set it to 1. Then, as in the previous step, add any needed custom fields to your wsdl file and perform a normal modify request.

Adding Records with Inline Custom Field Values

You can add records with inline custom field values.

1. Modify the wsdl file to include custom fields for the object you're working with. Use the field name and add "__c" to the end of the name. (Note that there are two underscores before the c.)
2. Specify custom fields inline in an object to be used in the array passed into the add method. See add method. See XML Tag Limitations.

Chapter 6 OpenAir Complex Types

The OpenAir SOAP API contains two complex types:

- OpenAir Complex Types - that describe business objects contained in the WSDL.
- Web Services Method Complex Types - that hold parameters and return values for Web Services calls.

This chapter provides the parent and children relationships of each OpenAir Complex Type. It includes a statement of its use and a listing of children. Links to supported calls are also provided.

See Web Services Method Complex Types to review Web services calls.

oaActualcost

Use this complex type to add or update actual cost information. oaActualcost has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
userid	The ID of the user.
date	Date for the actual cost.
externalid	If the record was imported from an external system, you store the unique external record ID here.
period	The time period of the actual cost: Daily, Weekly, Monthly, Quarterly, Annually.
created	Time the record was created.
currency	Currency of the cost field.
notes	Notes.
cost	The cost.
cost_typeid	The ID of the cost_type.
is_accrual	A 1/0 field indicating whether this actual cost is an accrual.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaAddress

Use this complex type to add or update address information. oaAddress has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
salutation	Contact's salutation
mobile	Mobile phone number
state	State
email	Email address
addr2	Address line 2
city	City
fax	Fax number
addr1	Address line 1
middle	Middle name
country	Country
first	First name
last	Last name
phone	Phone number
addr4	Address line 4
zip	Zip code
addr3	Address line 3

This complex type supports the following methods: add, createAccount, createUser, and modify

oaAgreement

Use this complex type to track money through projects and billings. oaAgreement has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
number	The agreement number.
date	The date of the agreement.
name	The name of the agreement.
active	A 1/0 field indicating whether this is an active agreement.
externalid	External ID.
total	The agreement total. Dated by the date field.

Field Name	Description
created	Time the record was created.
currency	Currency for the money fields in the record.
notes	Notes.
customerid	Customer ID.
updated	Time the record was last modified.
code	Optional accounting system code for integration with external accounting systems.
acct_date	The accounting period date of the agreement.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaAgreement_to_project

Use this complex type to create a many-to-many link between projects and agreements. oaAgreement_to_project has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
agreementid	The ID of the associated agreement.
customerid	The ID of the associated customer. Does not need to be input as it can be derived inline from project_id.
projectid	The ID of the associated project.
active	A 1/0 field indicating whether this is an active agreement.
created	Time the record was created.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaApproval

Use this complex type to store approval information for timesheets, expense reports, and proposals. oaApproval has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
cc	Email cc field
notes	Notes

This complex type supports the following method: submit.

oaApprovalProcess

Use this complex type to read approval process information. oaApproval has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name used for display in popups and lists.
updated	Time the record was last modified.
created	Time the record was created.

This complex type supports the read method.

oaAttachment

Use this complex type to specify information about task and proposal attachments and documents or folders. oaAttachment has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
file_name	The true attachment name, as provided by the user on upload.
locked_by	The ID of the user who uploaded the file, 0 if unlocked.
notes	Notes associated with the attachment.
created	Time the record was created.
workspaceid	The ID of the associated workspace.
base64_data	Base 64 encoded binary data of the actual attachment file.
updated	Time the record was last modified.
attachmentid	If non-zero, the attachment record associated with this attachment.
parentid	The attachment ID of our immediate ancestor. If zero/null, this is a top-level document/folder.
hash_name	The name of the file as stored on disk in our system. This is the relative path to the file from the document root directory.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaAttribute

Use this complex type as a table that describes an attribute. oaAttribute has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	Name of the attribute.
attribute_setid	ID of the associated attribute set.
updated	Time the record was last modified.
created	Time the record was created.
notes	Notes.

This complex type supports the read method.

oaAttributeset

Use this complex type to describe an attribute set. oaAttributeset has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	Name of the attribute.
created	Time the record was created.
updated	Time the record was updated.
notes	Attributeset notes

This complex type supports the read method.

oaBase

oaBase is a base object for most OpenAir Complex Types. Collections are passed as oaBase objects and most OpenAir Complex Types are derived from oaBase.

The following complex types use oaBase:

oaAddress	oaEstimatephase	oaPurchaser
oaAgreement	oaEvent	oaPurchaserequest
oaApproval	oaHierarchy	oaRatecard
oaBooking	oaHierarchyNode	oaReimbursement
oaBookingType	oaHistory	oaRequest_item
oaBudget	oaImportExport	oaResourceprofile
oaBudgetAllocation	oaInvoice	oaResourceprofile_type
oaCategory	oaItem	oaRevenue_recognition_rule
oaCcrate	oaJobcode	oaRevenue_recognition_rule_amount

oaCompany	oaLeave__rule	oaRevenue_recognition_transaction
oaContact	oaLeave_accrual_rule_to_user	oaSchedulerequest
oaCostcenter	oaLeave_accrual_transaction	oaSchedulerequest_item
oaCurrency	oaLoadedCost	oaSlip
oaCurrencyrate	oaModule	oaSlipstage
oaCustField	oaPayment	oaSwitch
oaCustomer	oaPaymentterms	oaTask
oaCustomerpo	oaPaymenttype	oaTaskTimecard
oaCustomerpo_to_project	oaPayrolltype	oaTaxLocation
oaCustomField	oaPreference	oaTaxRate
oaDate	oaProduct	oaTerm
oaDeal	oaProject	oaTicket
oaDealcontact	oaProjectbillingrule	oaTimecard
oaDealschedule	oaProjectbillingtransaction	oaTimesheet
oaDepartment	oaProjectlocation	oaTimetype
oaEntitytag	oaProjectstage	oaTodo
oaEnvelope	oaProjecttask	oaUprate
oaError	oaProjecttask_type	oaUser
oaEstimate	oaProjecttaskassign	oaUserWorkschedule
oaEstimateadjustment	oaProposal	oaVendor
oaEstimateexpense	oaProposalblock	oaWorkspacelink
oaEstimatelabor	oaPurchase_item	oaWorkspaceuser
oaEstimatemarkup	oaPurchaseorder	

oaBooking

Use this complex type to book a user to a project. oaBooking has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
hours	The number of hours booked to this project during this date range. This is either the actual booked hours or derived from the percentage.
ownerid	The ID of the associated user creating the booking.
userid	The ID of the associated user.
startdate	The start date of the booking.
percentage	The percentage of time booked to this project during this date range. This is either the actual booked percentage or derived from the hours.
projectid	The ID of the associated project.
externalid	If the record was imported from an external system you store the unique external record ID here.

Field Name	Description
booking_typeid	The ID of the associated booking_type.
project_task_id	The ID of the task within the associated project.
created	Time the record was created.
repeatid	The ID of the associated repeating event.
enddate	The end date of the booking.
notes	Booking notes.
customerid	The ID of the associated customer.
updated	Time the record was last updated or modified.
as_percentage	A 1/0 field indicating which of the fields (hours or percentage) are actual, and which is derived. 1 = percentage is actual and hours is derived. 0 = hours in actual and percentage is derived.
starttime	Start time.
endtime	End time.
job_codeid	The ID of the associated job code.
locationid	The location ID for this booking.
notify_owner	A 1/0 field indicating whether to send email to the requestor when the booking is modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaBookingType

Use this complex type to describe a booking type such as billable, non-billable, or business development used in Resources module bookings. oaBookingType has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
priority	The priority of the booking type (1 - 9).
created	Time the record was created.
notes	Booking notes.
name	The name of the booking type.
active	A 1/0 field specifying if the type is active.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaBudget

Use the oaBudget complex type to create a budget entry. oaBudget has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
date	The date of the budget entry.
name	The name.
projectid	The ID of the associated project.
total	The total value of budget entry. Dated by the date field.
budgetcategory_id	The ID of the budget category.
categoryid	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
notes	Budget notes.
customerid	The ID of the associated customer.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaBudgetAllocation

Use this complex type to allocate users and activity to a budget. oaBudgetAllocation has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
budgetid	The ID of the associated budget.
userid	The ID of the associated user.
date	The date of the budget entry.
projectid	The ID of the associated project.
budgetactivity_id	The ID of the budget activity.
total	The total value of budget entry. Dated by the date field.
budgetcategory_id	The ID of the budget category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
updated	Time the record was last modified.
allocation	The percentage of the budget entry that this user was allocated to.

This complex type supports the following methods: read, add, modify, and upsert.

oaCategory

Use this complex type for a service, category, activity or time type in the Proposals, Timesheets, and Invoices modules. oaCategory has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.
active	A 1/0 field indicating whether this is designated as an active customer.
taxable	A 1/0 field indicating whether this item is taxable, vat-taxable, etc.
externalid	If the record was imported from an external system you store the unique external record ID here.
other_rate_type	The time the other_rate field applies to. Valid entries are Day, Week, Month, Quarter, Year and Session.
other_rate	The rate for another time billing metric.
currency	Currency for the money fields in the record.
created	Time the record was created.
rate	The hourly billing rate.
cost_centerid	The ID of the associated cost center.
fixed_fee	The fixed fee value of this service.
updated	Time the record was last updated or modified.
code	Optional accounting system code for integration with external accounting systems.
notes	Category notes.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCategory_1

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_2, oaCategory_3, oaCategory_4, and oaCategory_5. oaCategory_1 has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.
active	A 1/0 field indicating whether this is designated as an active customer.
taxable	A 1/0 field indicating whether this item is taxable, vat-taxable, etc.
externalid	If the record was imported from an external system you store the unique external record ID here.

Field Name	Description
other_rate_type	The time the other_rate field applies to. Valid entries are Day, Week, Month, Quarter, Year and Session.
other_rate	The rate for another time billing metric.
currency	Currency for the money fields in the record.

This complex type supports the following methods: read, add, modify, and upsert. Use custom equal to when requesting custom fields.

oaCategory_2

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_3, oaCategory_4, and oaCategory_5. oaCategory_2 has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.
code	Optional accounting system code for integration with external accounting systems.
externalid	If the record was imported from an external system you store the unique external record ID here.
active	A 1/0 field indicating whether this is designated as an active customer.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Category notes_2

This complex type supports the following methods: read, add, modify, and upsert. Use custom equal to when requesting custom fields.

oaCategory_3

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_4, and oaCategory_5. oaCategory_3 has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.
code	Optional accounting system code for integration with external accounting systems.

Field Name	Description
externalid	If the record was imported from an external system you store the unique external record ID here.
active	A 1/0 field indicating whether this is designated as an active customer.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Category notes_3

This complex type supports the following methods: read, add, modify, and upsert. Use custom equal to when requesting custom fields.

oaCategory_4

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_3, and oaCategory_5. oaCategory_4 has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.
code	Optional accounting system code for integration with external accounting systems.
externalid	If the record was imported from an external system you store the unique external record ID here.
active	A 1/0 field indicating whether this is designated as an active customer.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Category notes_4

This complex type supports the following methods: read, add, modify, and upsert. Use custom equal to when requesting custom fields.

oaCategory_5

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_3, and oaCategory_4. oaCategory_5 has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The category name.

Field Name	Description
code	Optional accounting system code for integration with external accounting systems.
externalid	If the record was imported from an external system you store the unique external record ID here.
active	A 1/0 field indicating whether this is designated as an active customer.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Category notes_5

This complex type supports the following methods: read, add, modify, and upsert. Use custom equal to when requesting custom fields.

oaCcrate

Use this complex type to document the category customer rate table. oaCcrate has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the category this rate is associated with.
currency	The currency these rates are quoted in.
rate	The hourly billing rate.
created	Time the record was created.
notes	Notes about the table.
customerid	The ID of the customer this rate is associated with.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaCompany

Use this complex type to specify basic company information and the company switches. oaCompany has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
addr_state	State name.
addr_mobile	Mobile phone number.
VAT_registration_number	VAT registration number.

Field Name	Description
addr_country	Country name.
addr_phone	Phone number.
addr_addr4	Fourth line of the address.
hide_rate	Hide hourly rate from normal user types in the company.
updated	Time the record was last updated or modified.
company	The company name, as it should be printed on invoices, etc.
addr_zip	Zip code of the address.
addr_first	First name.
addr_email	Email address.
addr_addr3	Third line of the address.
nickname	The company nickname.
addr_addr1	First line of the address.
addr_last	Last name.
is_multicurrency	Multiple currencies.
currencies	The currencies for the money fields in the record.
businesstype	General business category.
addr_middle	Middle name.
addr_fax	Fax number.
created	The time the record was created.
addr_salutation	Salutation.
base_currency	Base currency.
addr_addr2	Second line of the address.
addr_city	City name.
rate_from	Billing rate is pulled from: category, user, customer/project, or user/project.
workscheduleid	The ID of the associated primary account workschedule. (read-only field)
flags	Company-specific flags.

This complex type supports the following methods: read, add, modify, and upsert. Also refer to oaSwitch for information on company-specific flags.

oaContact

Use this complex type to specify contact information. A contact is associated with a customer, particular client, or prospect company. oaContact has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
addr_state	State.
addr_mobile	Mobile phone number.

Field Name	Description
addr_country	Country.
customer_company	Import-only field to specify customer by company name.
job_title	The contact's job title.
addr_phone	Phone number.
addr_addr4	Fourth line of the address.
updated	Time the record was updated or modified.
can_bill_to	A 1/0 field indicating if the contact can be a billing contact.
addr_zip	Zip code.
addr_first	First line of the address.
code	Optional accounting system code for integration with external accounting systems.
addr_email	Email address.
addr_addr3	Third line of the address.
addr_addr1	First line of the address.
addr_last	The contact's last name.
name	The name of the contact. This will be automatically generated if not supplied.
active	A 1/0 field indicating an active contact.
externalid	If record is imported from external system, store the unique external record ID here.
can_sold_to	A 1/0 field indicating if the contact can be a sold to contact.
addr_middle	The contact's middle name.
addr_fax	Fax number.
created	Time the record was created.
addr_salutation	The contact's salutation.
addr_city	City.
addr_addr2	The second line of the address.
notes	Notes field.
customerid	The ID of the associated customer.
can_ship_to	A 1/0 field indicating if the contact can be a shipping contact.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCostcategory

Use this complex type to add or update cost category information. oaCostcategory has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.

Field Name	Description
active	A 1/0 field indicating if this cost category is active.
externalid	If the record was imported from an external system, you store the unique external record ID here.
created	Time the record was created.
notes	Notes.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaCostcenter

Use this complex type to specify cost center information. oaCostcenter has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
notes	Cost center notes.
name	The name of the cost center.
active	A 1/0 field indicating whether this is active.
updated	Time the record was last updated or modified.
externalid	If the record was imported from an external system you store the unique external record ID here.
code	Optional accounting system code for integration with external accounting systems.

This complex type supports the following methods: read, add, modify, and upsert.

oaCosttype

Use this complex type to add or update cost type information. oaCosttype has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
active	A 1/0 field indicating if this cost category is active.
externalid	If the record was imported from an external system, you store the unique external record ID here.

Field Name	Description
created	Time the record was created.
notes	Notes.
updated	Time the record was last modified.
cost_categoryid	The id of the associated cost category.

This complex type supports the following methods: read, add, modify, and upsert.

oaCurrency

Use the currency complex type to specify exchange rates that override market rates. oaCurrency has the following children.

Field Name	Description
rate	The account's custom conversion rate.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
symbol	The currency symbol.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaCurrencyrate

Use the currency rate complex type to read currency rates. oaCurrencyrate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
crate	The account's currency conversion rate.
csymbol	The currency symbol.
cname	The name of the currency rate.
date	The date of the rate.
type	Blank for rates with date filled in, otherwise: PAST - conversion rates for dates prior to the first date in the table FUTURE - conversion rate for dates in the future

This complex type supports the read method.

oaCustField

Use this complex type to retrieve metadata about custom fields such as name, association, and picker type. oaCustField has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
userid	The ID of the user who created or owns this custom field.
rows	The number of display rows for text area fields
size	The display size of the field on forms.
valuelist	A list of values for radio groups and popup menu fields in csv format.
required	A 1/0 field indicating if this field is required.
decpos	The decimal size of the field.
picker	The type of field for on screen representation: numeric, currency, date, text, textarea, check, radio, drop down, drop text, selector, or alloc_gr.
association	The association table or type of object this field is associated with. See Association Table or Type of Object for a list of associations.
updated	Time the record was last updated or modified.
seq	The sequence number of the field.
divider_text	Optional divider text.
maxlength	The maximum length of data in the field.
mover	A 1/0 field indicating if the selector should have mover controls.
name	The name of the custom field.
active	A 1/0 field indicating if this alert is active.
next_seq	Next sequence number to use.
description	The description of the custom field.
force_unique	A 1/0 field indicating if this field is unique.
defnow	A 1/0 field indicating if date fields default to today.
created	Time the record was created.
hint	The hint used on forms.
title	The title used on forms with this custom field.
divider	A 1/0 field indicating whether to paint a divider.
never_copy	A 1/0 field indicating if the field can be cloned.
hidden_data_entry	A 1/0 field indicating whether the custom field should be hidden on the data entry UI.

This complex type supports the following methods: read, add, modify, and upsert.

Association Table or Type of Object

The oaCustField complex type uses the following associations. The association is the name of the table that the custom field is related to. For more information, see association at the following URL: https://www.openair.com/database/single_user.html#cust_field.

accounts_payable	event	receiving
agreement	fulfillment	request_item
attachment	invoice	revenuerecognitionrule
authorization	item	revenue_container
authorization_item	issue	revenue_stage
booking	manufacturer	revenue_recognition_transaction
booking_request	payment_type	schedule_by_day
carrier	payroll_type	schedule_request
category	phase	schedule_request_item
contact	product	slip
cost_center	project	ticket
customer	projectbillingrule	timesheet
customerpo	project_task	timetype
deal	proposal	todo
deal_booking_request	purchase_item	user
department	purchaseorder	vendor
discussion	purchaser	workspace
envelope	purchaserequest	

oaCustomField

Use this complex type to specify custom fields. oaCustomField has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
value	Value of the field for a specific record.
name	The name of the custom field.
type	Association of the custom field.

This complex type supports the following methods: modify (with custom equal to attribute) and read (with custom equal to attribute).

oaCustomer

Use this complex type for customer, client or patient information. The customer is the individual or company that is billed or expensed. oaCustomer has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
addr_mobile	Customer's mobile phone number.
addr_country	Customer's country.
billing_addr_first	First name on the billing address.
billing_addr_middle	Middle name on the billing address.
billing_addr_addr2	Second line on the billing address.
invoice_layoutid	The ID of the associated invoice layout.
rate	Hourly billing rate for this customer.
bus_typeid	Type of business this customer is in.
billing_addr_zip	Zip code on the billing address.
contact_addr_mobile	Contact's mobile phone number.
code	Optional user-defined code.
tb_approver	The user_id of the invoice approver if this is a single approver process. This field is mutually exclusive with tb_approvalprocess. If -1 then the approver is the owners manager. If -2 then the approver is the owners manager's manager.
contact_addr_city	Contact's city.
addr_last	Customer's last name.
territoryid	The territory for this customer.
name	The nickname used for display in pop-up windows and lists.
billing_addr_fax	Fax number on the billing address.
hierarchy_node_ids	Comma delimited list - hierarchy nodes this object belongs to.
billing_addr_state	State on the billing address.
contact_addr_first	Contact's first name.
addr_fax	Customer's fax number.
addr_city	Customer's city.
hear_aboutid	How did they hear about us.
billing_addr_country	Country on the billing address.
statements	A 1/0 field indicating if this customer can view statements.
contact_addr_zip	Contact's zip code.
contact_addr_email	Contact's email address.
company_sizeid	This customer's company size.
web	Customer's Web address.
currency	Currency for the money fields in the record. Also the default currency when an invoice is created.
billing_addr_city	City on the billing address.
cost_centerid	The ID of the associated cost center.

Field Name	Description
contact_addr_addr4	Fourth line of the contact's address.
addr_zip	Customer's zip code.
contact_addr_addr2	Second line of the contact's address.
addr_addr1	First line of the customer's address.
contact_addr_addr3	Third line of the contact's address.
addr_middle	Customer's middle name.
addr_addr2	Second line of the customer's address.
contact_addr_last	Contact's last name.
notes	Notes about the customer.
contact_addr_phone	Contact's phone number.
tb_approvalprocess	The approvalprocess_id of the invoice approval process. This field is mutually exclusive with tb_approver.
primary_contactid	The billing contact ID.
contact_addr_middle	Contact's middle name.
billing_addr_addr3	Third line on the billing address.
addr_addr4	Fourth line of the customer's address.
contact_addr_state	Contact's state.
contact_addr_fax	Contact's fax number.
contact_addr_addr1	First line of the contact's address.
filterset_ids	Comma delimited list - filter sets this object belongs to.
addr_first	Customer's first name.
billing_addr_salutation	Salutation on the billing address.
addr_addr3	Third line of the customer's address.
active	A 1/0 field indicating whether this is designated as an active customer.
billing_contact_id	The billing contact ID.
externalid	If the record was imported from an external system you store the unique external record ID here.
addr_salutation	Customer's salutation.
contact_addr_salutation	Contact's salutation.
invoice_prefix	Text to start every invoice number with.
type	A C/P field indicating whether this is Customer or a Prospect.
billing_addr_addr4	Fourth line on the billing address.
addr_state	Customer's state.
contact_addr_country	Contact's country.
userid	The user ID of the customer or owner.
billing_addr_phone	Phone number on the billing address.
terms	Standard payment terms for the customer. Textual description like Net 30.
addr_phone	Customer's phone number.
created	Time the record was created.
invoice_text	Text to display on every invoice.

Field Name	Description
billing_addr_email	Email address on the billing address.
company	The company name.
addr_email	Customer's email address.
updated	Time the record was last updated or modified.
shipping_contactid	The shipping contact ID.
billing_addr_addr1	First line on the billing address.
billing_addr_last	Last name on the billing address.
billing_addr_mobile	Mobile phone number on the billing address.
sold_to_contact_id	The sold to contact ID.
createtime	Same as the created field (for legacy systems).
ta_include	A 1/0 field indicating whether a Timesheet filterset is applied.
te_include	A 1/0 field indicating whether an Expense Report filterset is applied.
updatetime	Same as the updated field (for legacy systems).
billing_code	The customer billing code. Used in bulk invoicing.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCustomerpo

Use this complex type to track money through projects and billings. oaCustomerpo has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
number	The customerpo number.
date	The date of the customerpo.
name	The name of the customerpo.
active	A 1/0 field indicating whether this is an active customerpo.
externalid	If the record was imported from an external system you store the unique external record ID here.
total	The customerpo total. Dated by the date field.
created	Time the record was created.
currency	Currency for the money fields in the record.
notes	Notes.
customerid	The ID of the associated customer.
updated	Time the record was last modified.
code	Optional accounting system code for integration with external accounting systems.
acct_date	The accounting period date of the customerpo.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCustomerpo_to_project

Use this complex type to create a many-to-many link between projects and customers. oaCustomerpo_to_project has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
customerpoid	The ID of the associated customerpo.
created	Time the record was created.
customerid	The ID of the associated customer.
active	A 1/0 field indicating whether this is an active customerpo.
updated	Time the record was last modified.
externalid	If the record was imported from an external system, you store the unique external record ID here.
projectid	The ID of the associated project.

This complex type supports the following methods: read, add, modify, and upsert.

oaDate

Use this complex type to specify date information. oaDate has the following children.

Field Name	Description
year	Year.
month	Month.
day	Day.
hour	Hour.
minute	Minute.
second	Second.

oaDeal

Use this complex type to specify a potential sale to a prospect or customer. A deal can also be associated with a contact, an estimate, a todo, or an event. oaDeal has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
closed	When this deal was closed.

Field Name	Description
stage	The % of the work complete for this deal.
userid	The ID of the associated user.
status	The status for this deal: O - Open, C - Closed, L - Lost
name	The name/description of the deal.
territoryid	The territory for this deal.
active	Is this record active?
rating	The rating for this deal.
created	Time the record was created.
opened	When this deal was first opened.
notes	Notes for this deal.
customerid	The ID of the associated customer.
exported	Date and time the record was marked as exported.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaDealcontact

Use this complex type to specify contact information for a deal. oaDealcontact has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
contactid	The related contact.
dealid	The deal ID.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaDealschedule

Use this complex type to specify schedule information for a deal. A deal, among other things, consists of a total deal amount and a potential closing date. However, this total amount can be broken down into smaller portions, each with its own potential closing date. A dealschedule is one of these smaller amount portions, and is associated with a particular deal. oaDealschedule has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
amount	The amount this portion of the deal is worth (in the currency of the deal). Dated by the date field.
dealid	ID of the deal associated with this deal portion.
date	The potential closing date for a deal portion.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaDepartment

Use this complex type to specify department information and associate a user with a department. oaDepartment has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The user ID of the head of the department.
notes	Notes about the department.
name	The name used for display in lists.
updated	Time the record was last updated or modified.
externalid	If the record was imported from an external system, you store the unique external record ID here.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaEntitytag

Use this complex type to specify entity tag information. oaEntitytag has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
default_for_entity	A 1/0 field indicating whether this is the default row for this entity.
userid	The ID of the associated user.
projectid	The ID of the associated project.

Field Name	Description
created	Time the record was created.
tag_group_attributeid	The ID of the associated tag_group_attribute.
end_date	End date for this entity_tag.
customerid	The ID of the associated customer.
updated	Time the record was last updated or modified.
start_date	Start date for this entity_tag.
tag_group_attribute_name	The name of the associated tag group attribute.
tag_group_id	The ID of the associated tag group attribute.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaEnvelope

Use this complex type to specify information about tickets in an envelope. Envelopes are used to group individual receipts into an expense report. oaEnvelope has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
totreimburse	The total amount of reimbursable expenses in the envelope.
advance	The amount of any cash advance on the envelope.
number	The envelope tracking number.
date	The date of the envelope.
userid	The ID of the associated user.
status	The status of the envelope: O - open S - submitted A - approved R - rejected
currency	The currency this envelope is in.
tottickets	The total number of tickets in the envelope.
trip_reason	The reason for the trip.
approver	The userid of the envelope approver.
date_start	Starting date of the envelope (only used with auto-naming).
updated	Time the record was last updated or modified.
date_end	The ending date of the envelope (only used with auto-naming).
name	The name of the envelope.
submitted	The date the envelope was submitted.
total	The total value of all the tickets in the envelope.

Field Name	Description
tax_locationid	Default tax location for this envelope.
created	Time the record was created.
approved	The date the envelope was approved.
balance	The outstanding balance on the envelope.
notes	Notes about the envelope.
is_overlapping	Read only flag returns is an envelope overlaps with another envelope.
externalid	If the record was imported from an external system, you store the unique external record ID here.
attachmentid	If non-zero, the attachment record associated with this envelope.
currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance as defined in database data definitions.
thin_client_id	Used by thin clients to reconcile imported records.
acct_date	The accounting period date of the envelope.

This complex type supports the following methods: read, add, modify, upsert, submit, and delete.

Note: There is an OpenAir internal switch that can be enabled to allow API editing of approved expense reports. To use this feature, open a support ticket and request that the following switch be enabled: API will allow editing of approved Expense reports. See Troubleshooting for instructions on how to create a support ticket.

oaError

Use this complex type to specify information about an error. oaError has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
comment	Additional comments.
text	Text of the error.
code	Error code returned by the API.

This complex type supports the read method.

oaEstimate

Use this complex type to specify estimate records for staffing, fixed costs, and discounts. It is used to create profit margin estimates for deals that are in the pipeline. oaEstimate has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
hide_expense	A 1/0 field indicating if expenses should be hidden in analysis report.
dealid	The ID of the associated deal.
name	The short description for the estimate.
created	Time the record was created.
notes	Notes about the estimate.
customerid	The ID of the associated customer.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimateadjustment

Use this complex type to specify estimate adjustment records. Estimate adjustments are the adjustment records associated with particular estimates. oaEstimateadjustment has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
amount	The amount of adjustment in money (in the currency of the estimate) or percentage of total expense or labor. The actual type is identified by amount_type field.
estimateid	The ID of the associated estimate.
name	The name for the estimate adjustment.
updated	Time the record was last updated or modified.
adjustment_type	A 1/0 field indicating the adjustment is for labor or expenses. If 1 - then adjustment is for labor. If 0 - then adjustment is for expenses.
amount_type	A 1/0 field indicating the type of the amount field. If 1 - then amount field represents percentage of time. If 0 - then amount field represents number of hours.

This complex type supports the read method.

oaEstimateexpense

Use this complex type to specify estimate expense records. oaEstimateexpense has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
estimateid	The ID of the associated estimate.
itemid	The ID of the associated expense item.
date	Date for the expense.
markup_type	A 1/0 field indicating the type of expense markup. If 1 - then use percentage of the cost. If 0 - then use the specific amount.
quantity	The quantity for the expense.
description	The short description for the estimate.
created	Time the record was created.
phaseid	The ID of the associated estimate phase.
markup	The amount of markup in percent or money as designated by markup_type field. Dated by the date field.
price	The cost of the expense. Dated by the date field.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimatelabor

Use this complex type to specify estimate staffing records. oaEstimatelabor has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
estimateid	The ID of the associated estimate.
loaded_cost	The loaded cost for the associated resource. Dated by the start_date field.
userid	The ID of the associated resource.
description	The short description for the estimate.
amount_type	A 1/0 field indicating the type of the amount field. If 1 - then amount field represents percentage of time. If 0 - then amount field represents number of hours.
created	Time the record was created.

Field Name	Description
amount	The number of hours or percentage of time associated with a given resource for a specific phase of an estimate. The actual type is identified by as_percentage field.
phaseid	The ID of the associated estimate phase.
end_date	End date for resource assignment.
billing_rate	The billing rate for the associated resource. Dated by the start_date field.
start_date	Start date for resource assignment.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimatemarkup

Use this complex type to specify information about phases for the estimate. oaEstimatemarkup has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
estimateid	The ID of the associated estimate.
percent	The percentage markup to add to the total expense amount.
total	The amount of expense (in the currency of the estimate) to use for this estimate in calculations.
created	Time the record was created.
phaseid	The ID of the associated estimate phase.
updated	Time the record was last updated or modified.
as_percentage	A 1/0 field indicating which expense markup to use: If 1 - then use percentage of the total, compute total markup. If 0 - then use the specific amount, compute percent markup.

This complex type supports the read method.

oaEstimatephase

Use this complex type to specify information about phases for the estimate. oaEstimatephase has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.

Field Name	Description
estimateid	The ID of the associated estimate.
name	The name for the estimate adjustment.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEvent

Use this complex type to specify information about events. An event is a historical record of an activity performed on behalf of a client or prospect. It could record the completion of a todo, the closing of a deal, or document a phone call or email message sent to a customer. oaEvent has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
contact_id	The ID of the associated contact.
userid	The ID of the user who created the event.
dealid	The ID of the associated deal.
name	The name or description of the event.
occurred	The date of the event.
created	Time the record was created.
notes	Notes related to the event.
updated	Time the record was last updated or modified.
customerid	The ID of the associated customer.

This complex type supports the following methods: read, add, modify, and upsert.

oaFieldAttribute

Use this complex type to supply additional attributes to all other complex types, with the exception of oaAddress, oaFieldAttribute, oaDate, and oaModule. You can use externalid and name fields as a foreign key, allowing you to read, add, createUser, modify, and upsert records in a single step instead of querying a record to first obtain the internal id.

Using an externalid field as a foreign key

1. Set the id field to the externalid field value instead of internal id field value.
2. Create a collection of oaFieldAttribute objects, one for each field that needs to be overridden.

3. Assign the collection to 'attributes' member of the target record. For code examples, refer to the following: Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

oaFieldAttribute has the following children.

Field Name	Description
name	Specifies the type of lookup: external - lookup the field using external_id field and name- lookup by the name field value.
value	A colon separated list consisting of two or three values. The value can either be external or name. Each is defined as follows: - External - matching record type to process lookup for - Name - field name as it exists in the object In addition, there can be an Optional flag 1 - instructs API to process comma separated list of values. This flag is not required for regular fields.

This complex type supports the following methods: read, add, createUser, modify, and upsert.

oaFilterset

Use this complex type to list names and IDs that define the table/id pairs to be filtered for each filterset. oaFilterset has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the filterset.
notes	Notes related to the filterset.
all_access	A 1/0 field indicating this filterset does not filter anything and cannot be deleted.
default_filter_set	A 1/0 field indicating whether this is the default new-user filterset.
active	A 1/0 field indicating whether this is designated as an active filter set.
externalid	If the record was imported from an external system, you store the unique external record ID here.
created	Time the record was created.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaForexInput

Use this complex type to allow multi-currency accounts to override historical and future currency conversion rates. oaForexInput has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
symbol	Currency symbol. Must be for one of the multiple currencies enabled in the account.
startdate	Optional start date for currency being set.
enddate	Optional end date for currency being set.
rate	Rate against the base currency for the account.
future	1 - if this is for future overrides. If used, start and end dates must be blank.
past	1 - if this is for past overrides. If used, start and end dates must be blank.
base	The currency symbol used as a base currency for the currency conversion table.
created	Date the record was created.
updated	Date the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

Note: There is an OpenAir internal switch that allows you to specify the rate against a user-defined currency. To use this feature, open a support ticket and request that the following switch be enabled: Enable user defined reporting currencies. See Troubleshooting for instructions on how to create a support ticket.

oaFormPermissionField

This complex type is for internal use only. oaFormPermissionField has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
form_name	Internal GUI form name
field_name	Internal GUI field name
hidden	A 1/0 field indicating whether this is to be hidden in the GUI.
readonly	A 1/0 field indicating whether this is to be readonly in the GUI
required	A 1/0 field indicating whether this is to be required in the GUI
default_value	Value to be prefilled when a new form is created.
save_and_create	List of field names prefilled with previous saved values.

This complex type supports the read method.

oaFulfillment

Use this complex type to specify information about the receipt of goods and services ordered by a purchase order. oaFulfillment has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
purchaseorder_id	Associated purchase order ID.
purchaserequest_id	Associated purchase request ID.
request_item_id	Associated request item ID.
carrier_id	Associated carrier ID.
slip_id	The ID of the associated slip if this expense was billed to a time bill.
purchase_item_id	Associated purchase item ID.
waybill_number	The waybill number.
date	Date of the fulfillment.
acct_date	The accounting period date of the fulfillment.
quantity	The quantity received.
notes	Fulfillment description notes.
created	Time the record was created.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaHierarchy

Use this complex type to specify hierarchy information. oaHierarchy has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
requireonform	A 1/0 field indicating whether this hierarchy should be added to the object type form.
name	The hierarchy name.
active	A 1/0 field indicating whether this is designated as an active hierarchy.
updated	Time the record was last updated or modified.
required	A 1/0 field indicating whether this hierarchy should be a required element on the object type form.
notes	Notes related to the hierarchy.
available_as_column	A 1/0 field indicating whether this hierarchy is available as a (customer, project or user) list column. Only one hierarchy per type can be displayed as a column in a list.
externalid	If the record was imported from an external system, you store the unique external record ID here.

Field Name	Description
primary_dropdown_filter	A 1/0 field indicating if this hierarchy is used as a drop-down filter.
primary_user_filterset	A 1/0 field indicating if this hierarchy determines filter set access for projects.
type	The type (table name) of the hierarchy: customer, project, or user.

This complex type supports the following methods: read, add, modify, and upsert.

oaHierarchyNode

Use this complex type to specify information about a hierarchy node. oaHierarchyNode has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
hierarchyid	The ID of the associated hierarchy.
levelid	The ID of the associated hierarchy level.
isalevel	A 1/0 field indicating if this node is a level.
created	Time the record was created.
recordid	The record ID if not a node.
name	The hierarchy name.
isanode	The name of the hierarchy node.
updated	Time the record was last updated or modified.
externalid	If the record was imported from an external system, you store the unique external record ID here.
parentid	The hierarchy_node ID of our immediate ancestor. If zero/null, is a top-level node.
notes	Notes related to the hierarchy node.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaHistory

Use this complex type to specify history events. oaHistory has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the user associated with this history event.
action	The approval action: S - Submittal, P - Pending, A - Acceptance, R - Rejection, U - unapproval.

Field Name	Description
notes	Notes associated with the history event.
envelopeid	The ID of the associated envelope.
date	The date associated with this history event.

For more information on its use, refer to the Example I. read equal to C# and Example I. read equal to Java. Note that the read method for oaHistory only works with the "equal to" retrieval method.

oaImportExport

Use this complex type to specify table and ID pairs corresponding to an external application. It can be used in conjunction with read and the not-exported filter to request records that have not been exported. oaImportExport has the following children.

Field Name	Description
id	Internal ID of the actual record (slip, task, etc.) in its native table.
attributes	A collection of additional attributes for this complex type.
application	String describing the application making the association.
exported	Time of the last export from OpenAir. Required on import.
type	Complex type name of the exported record: Slip, Task, Projectassign, etc. Refer to the Types table. Please note that these names are case sensitive.
externalid	External identifier for the application.
imported	Time of the last import to OpenAir. Required on import.

This complex type supports the following methods: read, add, modify, and upsert.

oaInvoice

Use this complex type to specify invoice information for the header. oaInvoice has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
draw_date	The date of the draw.
number	The invoice number.

Field Name	Description
status	The status of the invoice (EZ Invoice, emailed Invoice): 0 => Unknown 1 => Not Sent 2 => Viewed 3 => EZ Requested 4 => Rejected 5 => Sent 6 => EZ Sent 7 => Retracted
date	The date of the invoice.
terms	Payment terms for this invoice.
invoice_layoutid	The ID of the associated invoice layout.
credit_reason	The reason for the credit.
currency	The currency this invoice is in.
tax_state	The state tax total for the invoice. Dated by the date field.
tax_federal	The federal tax total for the invoice. Dated by the date field.
tax_gst	The GST tax for the invoice. Dated by the date field.
emailed	Date the user emailed the invoice. For invoices created before this field existed, this field is set to 1970-01-01 to flag it as an unknown value.
tax_pst	The PST tax for the invoice. Dated by the date field.
draw	The amount of any draw against retainer for this invoice. Dated by the draw_date field.
contactid	The contact id for this invoice.
shipping_contactid	The shipping contact id for this invoice.
approval_status	The approval status of the invoice, only used if invoice approvals are used: O - Open S - Submitted A - Approved R - Rejected
access_log	The mailing and access history of the invoice, such as when the customer accessed it.
credit	The amount of any credit against the invoice. Dated by the date field.
tax_hst	The HST tax for the invoice. Dated by the date field.
tax	The tax total for the invoice. Dated by the date field.
total	The invoice total. Dated by the date field.
updated	Time the record was updated.
balance	The outstanding balance on the invoice. Dated by the date field.
notes	Notes associated with the invoice.
paperrequest	Date the user requested that a paper invoice be mailed.
customerid	The ID of the associated customer.

Field Name	Description
accounting	A 1/0 field indicating if an invoice has been sent to an accounting partner.
acct_date	The accounting period date of the invoice.
externalid	If the record was imported from an external system, you store the unique external record ID here.
papersend	Date the paper invoice was actually mailed.
credit_rebill_status	Credit/Rebill status for the original invoice: C = Credit Initiated and R = Re-Bill.
original_invoiceid	The original invoice ID for credit invoices.
attachmentid	The ID of the associated attachment.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: If not all invoices are returned from an API request, determine whether the following OpenAir internal switch is enabled: API will allow editing of approved Invoices. Speak with your Professional Services Consultant or create a support ticket. See Troubleshooting for instructions on creating a support ticket.

oaInvoiceLayout

Use this complex type to read invoice layout information. oaInvoiceLayout has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name used for display in popups and lists.
updated	Time the record was last modified.
created	Time the record was created.

This complex type supports the read method.

oaIssue

Use this complex type to specify issue information. oaIssue has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
number	The issue number that increments by 1.
prefix	A static alphanumeric issue number prefix.
name	The name of the issue (Prefix + number).
owner_id	The ID of the associated user creating the issue.

Field Name	Description
description	A short description of the issue, a synopsis.
customer_id	The ID of the associated customer.
project_id	The ID of the associated project.
project_task_id	The ID of the task within the associated project.
issue_category_id	The ID of the associated issue category.
issue_status_id	The ID of the associated issue status.
issue_stage_id	The ID of the associated issue stage.
issue_severity_id	The ID of the associated issue severity.
issue_source_id	The ID of the associated issue source.
issue_notes	The description of the issue.
resolution_notes	The description of the resolution.
date	The date of the issue.
date_resolution_required	The date the issue is required to be resolved.
date_resolution_expected	The date the issue is expected to be resolved.
date_resolved	The date the issue was resolved.
user_id	The ID of the user assigned to the issue.
priority	The priority of the task (1 - 100).
attachment_id	If non-zero, the attachment record associated with this issue.
updated	Time the record was updated.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaIssueCategory

Use this complex type to specify information about the issue category. oaIssueCategory has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
name	The name of the issue category.
active	A 1/0 field indicating whether this issue category is active.
notes	Notes associated with the issue category.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueSeverity

Use this complex type to specify information about the issue severity. oaIssueSeverity has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
name	The name of the issue severity.
active	A 1/0 field indicating whether this issue severity is active.
notes	Notes associated with the issue severity.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueSource

Use this complex type to specify information about the issue source. oaIssueSource has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
name	The name of the issue source.
active	A 1/0 field indicating whether this issue source is active.
notes	Notes associated with the issue source.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueStage

Use this complex type to specify information about the issue stage. oaIssueStage has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.

Field Name	Description
name	The name of the issue stage.
default_for_new	A 1/0 field indicating whether this is the default stage for new issues.
considered_closed	A 1/0 field indicating whether issues in this stage are considered closed.
position	The position of the stage.
notes	Notes associated with the issue stage.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueStatus

Use this complex type to specify information about the issue status. oaIssueStatus has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the issue status.
active	A 1/0 field indicating whether this issue status is active.
created	Time the record was created.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaItem

Use this complex type to specify item information such as expense item, expense type, expense, inventory item. oaItem has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
cost	The default cost per unit of measure for the item. 3 decimal places to handle items like mileage at 32.5 cents.
active	A 1/0 field indicating whether this is designated as an active customer.
taxable	A 1/0 field indicating whether this item is taxable, VAT-able, etc.
externalid	If the record was imported from an external system you store the unique external record ID here.
unitm	The unit of measure for the item, i.e., EA.
currency	Currency for the money fields in the record.

Field Name	Description
cost_centerid	The ID of the associated cost center.
name	The item name.
type	The type of item. Add new types when type-specific information can be captured for the slip or ticket templated from this item: R - for regular item. M - for mileage item.
code	Optional accounting system code for integration with external accounting systems.
updated	Time the record was last updated or modified.
tp_cost	The policy threshold amount.
tp_comp	Ticket policy comparison: ge - greater than or equal to gt - greater than
tp_notes_required	Notes are required if the ticket triggers the policy.
tp_unit_or_total	The ticket policy is applied against: U - Unit price T - Total
tax_location_id	The ID of the associated tax location.
cost_is_fixed	A 1/0 field indicating whether the user is allowed to change the cost on a receipt.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaJobcode

Use this complex type to specify job code information. oaJobcode has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid_fte	The user ID of the FTE (Full Time Equivalent) generic resource.
loaded_cost	Loaded cost for this job code.
name	The name for the job code.
updated	Time the record was last updated or modified.
notes	Notes associated with the job code.
externalid	If the record was imported from an external system you store the unique external record ID here.
currency	Currency for the money fields in the record.

Field Name	Description
code	Optional accounting system code for integration with external accounting systems.
active	A 1/0 field indicating if this is an active job code.

This complex type supports the following methods: read, add, modify, and upsert.

oaLeave_accrual_rule

Use this complex type to specify leave accrual rule information. oaLeave_accrual_rule has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
project_task_filter	CSV list of project_tasks that will trigger a draw down.
category_filter	CSV list of categories that will trigger a draw down.
lose_how	How is accrued time lost: N - Never A - The users anniversary date Y - End of year
cap	Number of hours to cap the accrual at.
timetype_filter	CSV list of timetypes that will trigger a draw down.
project_filter	CSV list of projects that will trigger a draw down.
period	The period for the cap.
draw_down_when	Generate the draw down when: R - When leave accrual is run. A - When a timesheet is approved.
notes	Notes associated with the leave accrual rule.
active	A 1/0 field indicating whether this is an active billing rule.
name	The name for the leave accrual rule.
updated	Time the record was last updated or modified.
grace_days	How many days is the grace period before accrued time is lost.
timing	When the accrual is applied: S - start of the period. E - end of the period.
amount	The number of hours per period.

This complex type supports the following methods: read, add, modify, and upsert.

oaLeave_accrual_rule_to_user

Use this complex type to map leave accrual rules to users. oaLeave_accrual_rule_to_user has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the associated user.
end_date	The date the accrual rule stops applying to the user.
start_date	The date the accrual rule starts applying to the user. This is required.
updated	Time the record was last updated or modified.
transfer_balance_to	ID of leave_accrual_rule_to_user record where balance should be transferred to.
leave_accrual_ruleid	The ID of the associated accrual rule.

This complex type supports the following methods: read, add, modify, and upsert.

oaLeave_accrual_transaction

Use this complex type to specify leave accrual transaction information. oaLeave_accrual_transaction has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The date of the transaction.
taskid	The ID of the associated task if this is a draw down against a timesheet entry.
notes	Notes associated with the leave accrual transaction.
updated	Time the record was last updated or modified.
amount	The number of hours. A draw down must be a negative number. An accrual is typically a positive number but can be a negative number.
type	Indicates type of draw down: the type of the amount field. D - draw down. A - Accrual.
from_run	Indicates if this was generated from a run the leave accrual rules.
leave_accrual_ruleid	The ID of the associated accrual rule. This is a required field.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, and upsert.

oaLoadedCost

Use this complex type to specify loaded cost values for a user. oaLoaded Cost has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	ID of the user.
projectid	The ID if this loaded cost is associated with a specific project.
end	End date for the loaded cost for historical records.
updated	Time the record was last updated or modified.
currency	The currency of the cost field.
customerid	The ID of the associated customer.
current	A 1/0 field indicating if this is the current loaded cost record.
project_taskid	The ID if this loaded cost is associated with a specific project task. If this field is used, the project_id and customer_id must be empty.
cost	The fully loaded hourly cost of the user.
start	Start date for the loaded cost for historical records.
lc_level	If multiple loaded costs are used, this holds the level of loaded cost 0 - primary loaded cost 1 - secondary loaded cost 2 - tertiary loaded cost

This complex type supports the following methods: read, add, modify, and upsert.

oaModule

Use this complex type to specify Module availability. oaModule has the following children.

Field Name	Description
abbr	Abbreviation within OpenAir.
enabled	A 1/0 field indicating whether the module is enabled.

This complex type supports the read method.

oaPayment

Use this complex type to specify payment information. oaPayment has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The date of the payment.
invoiceid	The associated invoice ID if a payment against a specific invoice.
notes	Notes associated with the payment.
updated	Time the record was last updated or modified.
total	The payment total. Dated by the date field.
invoice_number	The associated invoice number if a payment against a specific invoice.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer if this is a retainer payment.
bulk_paymentid	The ID of the bulk_payment transaction if this payment is part of a bulk_payment.
externalid	If the record was imported from an external system you store the unique external record id here.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPaymentterms

Use this complex type to specify payment terms information. oaPaymentterms has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_terms	A 1/0 field indicating whether this is the default payment terms (used for Customers, Vendors, Invoices and POs).
name	The payment terms name.
notes	Notes associated with the payment terms.
updated	Time the record was last updated or modified.
active	A 1/0 field indicating where this is designated as an active shipping terms 1/0.

This complex type supports the following methods: read, add, modify, and upsert.

oaPaymenttype

Use this complex type to specify payment type information. Payment types are used to specify the payment methods for individual receipts. oaPaymenttype has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
active	A 1/0 field specifying if the type is active.
notes	Notes associated with the payment type.
name	The name of the payment type.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPayrolltype

Use this complex type to specify payroll type information. oaPayrolltype has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
active	A 1/0 field specifying whether this is an active payrolltype.
externalid	If the record was imported from an external system, you store the unique external record ID here.
notes	Notes associated with the payroll type.
name	The name of the payroll type.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPreference

Use this complex type to specify preference or setting information. oaPreference has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
name	The name for the preference.
updated	Time the record was last updated or modified.
group_name	Optional group name for the preference.

Field Name	Description
userid	If the preference is specific to a user, this will be the user ID.
setting	The preference data is stored in this field.

This complex type supports the following methods: read, add, modify, and upsert.

oaProduct

Use this complex type to specify product information. Products are used to create request items, which ultimately appear as line items on purchase orders. oaProduct has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
manufacturerid	The manufacturer of this product.
um	The unit of measure for the product, i.e., EA.
name	The name for the product. This shows up on all the product pop-up windows in the application.
updated	Time the record was last updated or modified.
currency	The currency this cost is quoted in.
code	Optional accounting system code for integration with external accounting systems.
manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
active	A 1/0 field indicating that this is active.
taxable	A 1/0 field indicating whether this item is taxable.
externalid	If the record was imported from an external system, you store the unique external record ID here.
vendor_sku	The preferred vendor's sku for this product.
vendor_id	The preferred vendor from whom to purchase this product.
notes	Notes associated with the product.
standard_cost	The current standard cost per unit of measure for the product. 3 decimal places to handle amounts like mileage at 32.5 cents.

This complex type supports the following methods: read, add, modify, and upsert.

oaProject

Use this complex type to specify project information and to create one project from another project. Indicate the rules and settings to copy. oaProject has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
user_filter	Also allow these users to edit the project if the only_owner_can_edit switch is on.
message	Dashboard message.
auto_bill	A 1/0 field, 1 if the project can be auto-billed.
az_approver	The user_id of the project expense authorization approver if this is a single approver process. This field is mutually exclusive with az_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
po_approver	The user_id of the project purchase order approver if this is a single approver process. This field is mutually exclusive with po_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
auto_bill_cap	A 1/0 field, 1 if the project should have a cap on auto-billings.
invoice_layoutid	The ID of the associated invoice layout.
category_filter	A category (service) filter. This will hold a list of the categories that are allowed to book time to this project.
rate	The hourly billing rate.
notify_assignees	A 1/0 field indicating whether to send email to assigned users whenever a task in this project is added, modified, or deleted.
sync_workpace	A 1/0 field indicating whether to keep project resources in sync with linked workspace members.
notify_owner	A 1/0 field indicating whether to send email to the project owner when an ownership change is made.
br_approvalprocess	The approvalprocess_id of the project booking request approval process. This field is mutually exclusive with br_approver.
te_approver	The user_id of the project expense report approver if this is a single approver process. This field is mutually exclusive with te_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
ta_approvalprocess	The approvalprocess_id of the project timesheet approval process. This field is mutually exclusive with ta_approver.
te_approvalprocess	The approvalprocess_id of the project expense report approval process. This field is mutually exclusive with te_approver.

Field Name	Description
auto_bill_cap_value	The auto-billings cap amount (in the currency of the project).
updated	Time the record was last updated or modified.
code	Optional system code for integration with external accounting systems.
tb_approver	The user_id of the project invoice approver if this is a single approver process. This field is mutually exclusive with tb_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this project.
tax_location_name	Name of the tax location.
active	A 1/0 field indicating an active project.
name	The project name. This shows upon all the project pop-up windows in the application.
hierarchy_node_ids	The ID of the hierarchy node for this project.
externalid	If the record was imported from an external system, you store the unique external record ID here.
po_approvalprocess	The approvalprocess_id of the project purchase order approval process. This field is mutually exclusive with po_approver.
project_stageid	The ID of the project stage.
tax_locationid	The ID of the associated tax location.
ta_approver	The user_id of the project timesheet approver if this is a single approver process. This field is mutually exclusive with ta_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
pr_approver	The user_id of the project purchase request approver if this is a single approver process. This field is mutually exclusive with pr_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
locationid	The location ID for this project (DEPRECATED).
finish_date	The calculated finish date of the project.
msp_link_type	If imported from Microsoft project, this field describes the state: "" not imported from MSP 'I' imported and locked for edit 'U' imported but unlocked for edit
customerid	The ID of the associated customer.

Field Name	Description
customer_name	The customer's name.
only_owner_can_edit	A 1/0 field indicating whether only the project owner can edit this project.
br_approver	The user_id of the project booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
userid	The user ID of the project owner.
az_approvalprocess	The approvalprocess_id of the project expense authorization approval process. This field is mutually exclusive with az_approver.
project_locationid	The location ID for this project.
budget	The budgeted revenue for the project.
currency	Currency for the money fields in the record.
cost_centerid	The ID of the associated cost center.
sga_labor	The allocated cost (SG and A) overhead percentage to apply to labor for profitability analysis.
invoice_text	Text to display on every invoice.
budget_time	The budgeted amount of time for the project, in hours.
start_date	The scheduled starting date of the project.
pr_approvalprocess	The approvalprocess_id of the project purchase request approval process. This field is mutually exclusive with pr_approver.
billing_contactid	The billing contact ID if different than the customer designated billing contact.
billing_code	The project billing code. Used in bulk invoicing.
created	Time the record was created.
no_dirty	A 1/0 field, 1 if we want this project to be marked dirty when it has finished the current recalc.
notes	Notes associated with this project.
create_workspace	A 1/0 field, 1 if an associated workspace is automatically created by the API. The project owner becomes the workspace owner.
tb_approvalprocess	The approvalprocess_id of the project invoice approval process. This field is mutually exclusive with tb_approver.
auto_bill_override	A 1/0 field, 1 if the project overrides the global auto_billing settings. The auto_bill table will hold the settings for the project.
template_project_id	ID of the project from which tasks and phases, billing rules, revenue recognition rules and other items will be copied.
copy_revenue_recognition_rules	Duplicates revenue recognition rules. A 1/0 field, 1 if the recognition rules should be copied.
copy_revenue_recognition_auto_settings	Duplicates revenue recognition rules auto-run settings. A 1/0 field, 1 if the auto-run settings should be copied.
copy_project_billing_rules	Duplicates project billing rules. A 1/0 field, 1 if the billing rules should be copied.

Field Name	Description
copy_project_billing_auto_settings	Duplicates project auto-bill settings. A 1/0 field, 1 if the auto-bill settings should be copied.
copy_project_pricing	Duplicates project pricing information. A 1/0 field, 1 if the project pricing information should be copied.
copy_custom_fields	Duplicates custom fields. A 1/0 field, 1 if the custom fields should be copied.
copy_loaded_cost	Duplicates project pricing information. A 1/0 field, 1 if the loaded costs should be copied.
copy_approvers	Duplicates project approvers. A 1/0 field, 1 if the project approvers should be copied.
copy_issues	Duplicates issues. A 1/0 field, 1 if the issues should be copied.
copy_notification_settings	Duplicates notification settings. A 1/0 field, 1 if the notification settings should be copied.
copy_dashboard_settings	Duplicates dashboard settings. A 1/0 field, 1 if the dashboard settings should be copied.
copy_invoice_layout_settings	Duplicates invoice layout settings. A 1/0 field, 1 if the invoice layout settings should be copied.
pm_approver_1	The user_id of the project approver 1 that is substituted into the approval processes. If -6 then the approver is the 1st additional project approver.
pm_approver_2	The user_id of the project approver 2 that is substituted into the approval processes. If -7 then the approver is the 2nd additional project approver.
pm_approver_3	The user_id of the project approver 3 that is substituted into the approval processes. If -8 then the approver is the 3rd additional project approver.
payroll_type_filter	A payroll type filter. This holds a list of the payroll types that are allowed to book time to this project.
shipping_contact_id	The shipping contact ID if different than the customer designated shipping contact.
sold_to_contact_id	The sold to contact ID if different than the customer designated sold to contact
filterset_ids	A comma separated list of filter set IDs this record should be part of.
attachmentid	If non-zero, the attachment record associated with this project.
rv_approver	The user_id of the project revenue_container approver if this is a single approver process. This field is mutually exclusive with rv_approvalprocess If -1 then the approver is the owners manager If -2 then the approver is the owners manager's manager If -3 then the approver is the project owner If -4 then the approver is self
rv_approvalprocess	The approvalprocess_id of the project revenue_container approval process. This field is mutually exclusive with rv_approver.
portfolio_projectid	The ID of the associated portfolio project.
is_portfolio_project	A 1/0 field - 1 if the project is a portfolio project.
copy_revenuerecognition_auto_settings	Duplicate of copy_revenue_recognition_auto_settings.

Field Name	Description
filtersetids	A comma separated list of filter set IDs this record should be part of.
notify_issue_assigned_to	A 1/0 field indicating whether to send email to a user whenever assigned to an issue.
notify_issue_closed_assigned_to	A 1/0 field indicating whether to send email to the assigned user whenever an issue is moved to a considered closed issue stage.
notify_issue_closed_customer_owner	A 1/0 field indicating whether to send email to the customer owner whenever an issue is moved to a considered closed issue stage.
notify_issue_closed_project_owner	A 1/0 field indicating whether to send email to the project owner whenever an issue is moved to a considered closed issue stage.
notify_issue_created_customer_owner	A 1/0 field indicating whether to send email to the customer owner whenever an issue is created.
notify_issue_created_project_owner	A 1/0 field indicating whether to send email to the project owner whenever an issue is created.
notify_sr_submitted_project_owner	A 1/0 field indicating whether to send email to the project owner when a schedule request is submitted for a user booked or assigned to the project.
ta_include	A 1/0 field indicating whether a Timesheet filterset is applied.
te_include	A 1/0 field indicating whether an Expense Report filterset is applied.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: To create one project from another project, use the add method. Use the template_project _id field to designate the ID of the project from which project data will be copied. Use a "1" to indicate the settings and rules you want copied. Available fields begin with copy_.

oaProjectassign

Use this complex type for the assignment by project feature to track users assigned to a project. oaProjectassign has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
user_id	The ID of the user assigned to this task.
project_groupid	The ID of the project group if the user was assigned as part of a project group.
updated	Time the record was last updated or modified.
job_codeid	The ID of the associated job code.
customer_id	The ID of the associated customer.
project_id	The ID of the project to which this user is assigned.
allocation	The percentage of time the associated user is allocated to this task.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectbillingrule

Use this complex type to specify project billing rules. oaProjectbillingrule has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
name	Name of this project billing rule.
user_filter	CSV list of users to limit the rule to.
cap_hours	The number of hours to cap the billing at for a time billing rule.
backout_gst	If they are using GST/HST/PST taxes, back out the GST/HST taxes from re-billed expenses.
ticket_maximums	Holds data on ticket maximums per expense type.
markup_type	A field indicating the type of expense markup: P - percentage of the cost. S - specific amount.
percent	The percentage value for a fixed fee percent trigger.
end_milestone	The ID of the ending milestone (project_task).
daily_roll_to_next	If the period cap is hit move the remainder to the next rule.
category_filter	CSV list of categories to limit the rule to.
exclude_non_reimbursable	Exclude non-reimbursable expenses.
percent_how	If the fixed fee is triggered by a percent complete, this holds how it is triggered: A - % complete of planned hours for the project B - % complete of planned hours for a phase or task (the task ID is held in the start_milestone field)
adjust_if_capped	If a transaction will exceed the cap, should it be adjusted to fit under the cap.
slip_stageid	The ID of the slip stage to assign to the transaction.
markup_category	The ID of the category a markup on expense receipts should be assigned to.
timetype_filter	CSV list of timetypes to limit the rule to.
cap	The amount to cap total billing for this rule at (in the currency of the project).
daily_cap_period	Period for the cap: D - day W - week M - month Q - quarter Y - year
active	A 1/0 field indicating whether this is an active billing rule.

Field Name	Description
description	The rule description.
categoryid	The ID of the category to assign to the transaction if it doesn't have a category.
start_milestone	The ID of the starting milestone (project_task).
end_date	End date of the rule.
rate_from	Where we get the rate from: U - Users R - Rate cards C - Category
type	The type of the billing rule: T - time billing rule. E - expense billing rule. F - fixed fee billing rule. P - purchase billing rule.
agreementid	The ID of the associated agreement.
customerpoid	The ID of the associated customerpo.
item_filter	CSV list of items to limit the rule to.
position	The position of the rule (0,1,2 etc.). Rules are evaluated in order and evaluation stops once a rule is satisfied.
rate_multiplier	Optional multiplier to adjust the time billing rate by.
project_task_filter	CSV list of tasks to limit the rule to.
rate_cardid	The ID of the associated rate card if using rate cards.
product_filter	CSV list of products to limit the rule to.
currency	Currency for the money fields in the record.
repeatid	The ID of the associated repeating event.
exclude_archived_ts	Exclude time from archive timesheets in time billing rules.
exclude_non_billable	Exclude non-billable expenses.
exclude_non_billable_task	Exclude non-billable tasks.
markup	The amount of markup in percent or monetary amount as designated by markup_type field.
start_date	Start date of the rule.
category_when	When the category be applied: N - Use the selected category if the time entry does not have a category. A - Always use the selected category.
projectid	The ID of the associated project.
stop_if_capped	If a transaction is not billed because it exceeds the cap, should the billing stop for this transaction.
amount	The amount for a fixed fee rule.
round_rules	Rules for rounding time.
daily_cap_is_per_user	Is the daily cap on a per user basis.

Field Name	Description
acct_date	The accounting period date to assign to the transaction.
acct_date_how	The accounting period date of the transaction is determined by: N - none, clear the value E - the entity (no change) C - container of the entity if available (i.e., timesheet, envelope) S - submitted date of the container A - approved date of the container M - set by the specified accounting date P - set by the specified accounting period
accounting_period_id	The ID of the associated accounting period.
daily_cap_hours	The number of hours to cap the period billing per user at.
cost_center_id	The ID of the associated cost center.
notes	Notes associated with this project billing rule.
customer_id	The ID of the associated customer.
daily_rate_multiplier	Optional daily multiplier to adjust the time billing rate by. This is a comma delimited list of the multipliers for the days of the week starting with Monday and ending with Sunday.
job_code_filter	CSV list of filters to limit the rule to.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id.
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectbillingtransaction

Use this complex type to specify project billing transactions. oaProjectbillingtransaction has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Notes associated with this project billing rule transaction.
hour	The number of hours for a T type.
userid	The ID of the associated user.
date	The date of the transaction.

Field Name	Description
taskid	The ID of the associated task
um	The unit of measure for an E or P type.
rate	The hourly rate for a T type. Dated by the date field.
slipid	The ID of slip that was created.
ticketid	The ID of the associated ticket.
project_taskid	The ID of the associated project task.
project_billing_ruleid	The ID of the associated project billing rule.
cost	The cost per unit of measure for an E type. The fixed price for an F type. Dated by the date field.
itemid	The ID of the associated item.
quantity	The quantity for an E or P type.
projectid	The ID of the associated project.
description	Description associated with billing rule transaction.
total	The total currency value. Dated by the date field.
categoryid	The ID of the associated category.
minute	The number of minutes for a T type.
type	The type of the transaction. Matches the type field in project_billing_rule.
job_codeid	The ID of the associated job code.

This complex type supports the read method.

oaProjectgroup

Use this complex type to document users who are assigned to a project task as a group. oaProjectgroup has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
assigned_users	The users assigned to this project group. Can be a comma delimited list.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Notes associated with the project group.
name	The name for the project group.
active	A 1/0 field indicating whether this is active.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectlocation

Use this complex type to specify project location information. oaProjectlocation has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Notes associated with the project location.
name	The name for the project location.
active	A 1/0 field indicating whether this is active.

This complex type supports the read method.

oaProjectstage

Use this complex type to specify project stage information. oaProjectstage has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Notes associated with the project stage.
name	The name of the project stage.
position	The position of the stage.
enable_team	A 1/0 field indicating if this should be the default stage for new projects.
enable_utilization	Is the utilization enabled at this stage.
enable_project_assignments	Are project level assignments enabled at this stage.
enable_phase_and_task	Are phases and tasks enabled at this stage.
enable_analysis	Is financial analysis enabled at this stage.
enable_billing	Is the project billing tab enabled at this stage.
enable_recognition	Is the recognition tab enabled at this stage.
enable_pricing	Is project pricing enabled at this stage. Off by default.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaProjecttask

Use this complex type to specify information about the individual tasks or work packages that comprise a project. oaProjecttask has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
notes	Notes associated with the project task.
name	Short description of this task.
priority	The priority of the task (1 - 9).
percent_complete	This field is an estimate of the percentage of planned time which has been completed. It has no relation to the actual time spent on a task. (A 5-hour task could consume 50 hours of work but still be only 25% complete.)
task_budget_cost	If task budgeting is enabled, this is the total cost of the task.
is_a_phase	A 1/0 field indicating if any other project_tasks have us as a parent.
seq	The sequence number of this task.
timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this task.
non_billable	If set to 1, this is not billable. This is only applicable for project billing rules.
externalid	If the record was imported from an external system you store the unique external record ID here.
default_category	The category to assign to a timesheet entry assigned to this task. The feature has to be enabled for this assignment to work.
all_can_assign	Is everyone able to assign time/expenses to this task.
predecessors	Comma delimited list of task IDs which must complete before this task can start.
customer_name	The name of the associated customer.
parentid	The task ID of our immediate ancestor. If zero or null, this is a project-level (top-level) task or phase.
projecttask_typeid	The ID of the associated project task type. Not for phases.
calculated_finishes	Calculated finish date.
predecessors_lag	Comma delimited list for task ID:days of lag time for predecessors. Only populated if there is a lag time.
currency	Currency for the money fields in the record. This should be the same as the project currency.
cost_centerid	The ID of the associated cost center
calculated_starts	Calculated start date of the project task.
estimated_hours	If the use task estimating feature is turned on, this field will have the estimated total time the task will take to complete. If zero, no estimating has happened so the estimate is the same as the plan.

Field Name	Description
project_name	The name of the associated project.
id_number	User-defined task ID.
closed	A 1/0 field indicating if this is closed task. Additional time can not be booked against closed task.
task_budget_revenue	If task budgeting is enabled this is the total projected billing for the task.
planned_hours	Total number of hours the task is estimated to require. This is the total amount of time the task should take if worked on continuously by one person with no interruptions. A task with zero planned hours is also known as a milestone.
use_project_assignment	Flag set to 1 if they are using the project level user assignment.
projectid	The ID of the associated project.
assign_user_names	A comma separated list of user nicknames to assign this task to. (project_task_assign object types can also be used.)
starts	Optional scheduled starting date of this task. Overrides computed date Start_date.
fnlt_date	The finish no later than date of the task. The task must be finished by this date.
customerid	The ID of the associated customer.
predecessors_type	Comma delimted list of task ID:relationship type for predecessors. Only populated if the relationship type is not finish-to-start.
default_category_1	A feature, if enabled, would assign this default_category_1 to the category_1 for many transactions that have a category_1_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_1 defined.
default_category_2	A feature, if enabled, would assign this default_category_2 to the category_2 for many transactions that have a category_2_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_2 defined
default_category_3	A feature, if enabled, would assign this default_category_3 to the category_3 for many transactions that have a category_3_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_3 defined.
default_category_4	A feature, if enabled, would assign this default_category_4 to the category_4 for many transactions that have a category_4_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_4 defined.
default_category_5	A feature, if enabled, would assign this default_category_5 to the category_5 for many transactions that have a category_5_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_5 defined.
manual_task_budget	If set to 1 then the task budget is manually entered rather than calculated by the system.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaProjecttask_type

Use this complex type to specify information about a project task type. oaProjecttask_type has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
notes	Notes associated with the project task type.
active	A 1/0 field specifying if the type is active.
name	The name of the projecttask_type.
updated	Time the record was last updated or modified.
suppress_notification	Suppress task notifications for this project task type.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjecttaskassign

Use this complex type to specify the list of users assigned to each task. oaProjecttaskassign has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the user assigned to this task.
planned_hours	The hours for this user if the planned hours at the user level feature is enabled.
updated	Time the record was last updated or modified.
projecttaskid	ID of the project task to which this user is assigned
externalid	If the record was imported from an external system you store the unique external record ID here.
project_groupid	The ID of the project group if the user was assigned as part of a project group.
projecttaskid	The ID of the project_task to which this user is assigned.
allocation	The percentage of time the associated user is allocated to this task.
job_codeid	The ID of the associated job code.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaProposal

Use this complex type to specify proposal information. oaProposal has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
number	The proposal number.
userid	The ID of the associated user.
status	The status of the proposal: D - Draft M - Submitted P - Approved Q - Rejected S - Sent V - Viewed A - Accepted R - Refused
attachments	If non-zero, the attachment record associated with this proposal. attachment_id
responded	The date and time the client accepted or refused.
sent	The date and time the proposal was delivered to the client.
access_log	The mailing and access history of the proposal.
response	Client response notes.
name	The name of this proposal.
submitted	The date and time the proposal was submitted for approval.
approved_by	The ID of the user who approved this proposal.
projectid	The ID of the associated project.
description	The description of this proposal.
total	The total amount. Dated by the currency_date field.
approved	The date and time the proposal was approved.
viewed	The date and time the client first viewed the proposal.
notes	Notes associated with this proposal.
customerid	The ID of the associated customer.
created_by	The ID of the user who created this proposal.
expires	The date the proposal is valid until.

This complex type supports the read method.

oaProposalblock

Use this complex type to specify proposal blocks, the blocks of text that a proposal is composed of. A block can be free form text or it can be associated with a template. If associated with a

template, it is updated when the template is updated. oaProposalblock has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
hour	The number of hours for a T block.
templateid	The ID of the associated template.
content	The content of the template.
um	The unit of measure for an E block or the rate description for an O block.
rate	The hourly rate for a T block. Dated by the currency_date field.
proposalid	The ID of the associated proposal.
slipid	The ID of the associated slip if this block was billed to TB.
seq	The sequence number of the block.
cost	The cost per unit of measure (in the currency of the proposal) for an E block, the billing rate for an O block, or the fixed price for a F block. Dated by the currency_date field.
itemid	The ID of the associated item.
name	The name of the this proposal block.
quantity	The quantity for an E block or an O block.
total	The total value of the block. Dated by the currency_date field.
description	The description of this proposal.
categoryid	The ID of the associated category.
minute	The number of minutes for a T block.
type	The type of the slip: X - is a text only block T - is an hourly rate block E - is an expense block F - is a flat price block O - is an other type block P - is a product block

This complex type supports the read method.

oaPurchase_item

Use this complex type to specify purchase item information, a single entry in a purchase order. oaPurchase_item has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
date	The date of the purchase item. The same as the purchase order date.
um	The unit of measure for the product, i.e., EA.
purchaserid	The ID of the purchaser or purchasing agent. This is always the same as the purchase order creator (purchaser_id).
attachmentid	If non-zero, the attachment record associated with this purchase item.
approved_cost	A snap-shot of the approved cost from the request item (in the currency of the purchase order). 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
cost	The cost per unit of measure at which the product is ordered (in the currency of the purchase order). 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
tax_location_name	The name of the tax location.
non_po	A 1/0 field indicating that this purchase item was created without a purchase order.
name	The purchase name.
total	The total value of the purchase (in the currency of the purchase order). Dated by the date field.
quantity_payable	The quantity that is payable.
cusomerid	The ID of the associated customer.
request_itemid	The ID of the associated request item.
vendor_quote_number	The vendor's quote number.
userid	The ID of the requester.
purchaserequestid	The ID of the associated purchase request.
manufacturerid	The ID of the associated manufacturer.
currency	Currency for the money fields in the record.
quantity_fulfilled	The quantity that has been fulfilled.
date_fulfilled	The date on which all of the quantity was fulfilled.
purchaseorderid	The ID of the associated purchase order.
allow_vendor_substitution	A 1/0 field indicating whether the vendor may be substituted.
order_reference_number	Unique reference number within purchase order.
total_with_tax	The total value of the purchase (in the currency of the purchase order)including tax. Dated by the date field.
quantity	The quantity of product_id for this purchase.
projectid	The ID of the associated project.
vendor_sku	The vendor's sku for this product.

Field Name	Description
vendorid	The ID of the associated vendor
notes	Notes associated with this purchase order block.
productid	The ID of the associated product.
acct_date	The accounting period date of the purchase item.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: There are several limitations regarding purchase items: Only short-order purchase items can be added to OpenAir and only project/customer combinations can be updated on a non short-order purchase item (switch enabled). See details as follows:

- For the capability to add short-order purchase items in OpenAir, go to Administration > Application Settings > Purchases Settings and select Other settings. Scroll down and select the check box to Enable the ability to create non-PO purchase items. These are purchase items for purchases made without an OpenAir PO.
- For the capability to modify non short-order purchase items, submit a support ticket and request that the following switch be enabled: API can modify purchase items' project association even when associated with a PO. Associated request items will also be updated. See Troubleshooting for instructions on how to create a support ticket.

oaPurchaseorder

Use this complex type to specify information about a purchase order. oaPurchaseorder has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
receivingid	The receiving location for this purchase order.
carrierid	The carrier to be used for shipping. Ship Via.
date	The date of the purchase order.
date_required	The date the purchase items on this purchase order are required.
attachmentid	If non-zero, the attachment record associated with this purchase order.
date_shipped	The date the materials were shipped if known.
date_expected	The date the materials are expected if known.
date_submitted	The date the purchase order was submitted.
name	The name of the purchase order (Prefix + number).

Field Name	Description
total	The purchase order total cost. Dated by the date field.
description	The description or purpose for the purchase order.
locationid	The F.O.B. location_id (DEPRECATED).
shipping_cost	The cost of shipping, if known. Dated by the date field.
shipping_termsid	The id of the associated shipping payment terms, indicating how the shipping costs will be charged.
accounts_payableid	The accounts payable location for this purchase order.
number	The purchase order number that increments by 1.
userid	The id of the user creating the purchase order. The purchasing agent.
terms	Payment terms for this purchase order.
total_purchase_items	The total number of purchase items in the purchase order.
currency	The currency this purchase order is in.
quantity_fulfilled	The quantity fulfilled on all the purchase items in this purchase order.
date_approved	The date the purchase order was approved.
date_order_placed	The date the purchase order was placed with the vendor.
date_fulfilled	The date on which all of the total quantity was fulfilled.
auto_track_payable_ with_fulfilled	A 1/0 field indicating that payability of quantities of items on this purchase order track automatically and directly with the fulfillment of those items.
total_quantity	The total quantity of all the purchase items in this purchase order.
purchase_items_fulfilled	The total number of fulfilled purchase items in the purchase order.
approval_status	The approval status of the purchase request: O - open P - pending approval A - approved R - rejected
vendorid	The id of the associated vendor that the purchase order is for.
notes	Notes to print on the purchase order.
ship_complete_only	A 1/0 field indicating that full order must ship together.
prefix	A static alphanumeric purchase order number prefix.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPurchaser

Use this complex type to specify information about a user who creates purchase orders. oaPurchaser has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
receivingid	The default receiving location for this purchaser.
userid	The ID of the associated user.
name	The name of the purchaser.
updated	Time the record was last updated or modified.
carrierid	The default carrier to be used for shipping. Ship Via.
notes	Notes associated with the purchaser.
exported	Date and time the record was marked as exported.
accounts_payableid	The default accounts payable location for this purchaser.
ship_complete_only	The default for the 1/0 field indicating that full order must ship together.
active	A 1/0 field indicating where this is designated as an active receiving location.

This complex type supports the read method.

oaPurchaserequest

Use this complex type to specify purchase request information. oaPurchaserequest has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
number	The purchase request number that increments by 1.
date	The date of the purchase request.
userid	The ID of the associated user creating the purchase request. The requester.
request_items_fulfilled	The total number of fulfilled request items in the purchase request.
total_request_items	The total number of request items in the purchase request.
ordered_request_items	The total number of request items on the purchase request which are part of a purchase order.
date_required	The date the material on this purchase request is required.
attachmentid	If non-zero, the attachment record associated with this purchase request.

Field Name	Description
currency	The currency of the total field.
quantity_fulfilled	The quantity fulfilled on all the request items in this purchase request.
date_approved	The date the purchaserequest was approved.
date_fulfilled	The date on which all of the total quantity was fulfilled.
total_quantity	The total quantity of all the request items in this purchase request
date_submitted	The date the purchaserequest was submitted.
approval_status	The approval status of the purchase request: O - open P - pending approval A - approved R - rejected
name	The name of the purchaserequest (Prefix + number).
projectid	The ID of the associated project that the material on this purchase request is for.
description	The description or purpose for the purchaserequest.
total	The purchase request total cost. Dated by the date field.
notes	Notes associated with the purchase request.
customerid	The ID of the associated customer that the material on this purchase request is for.
exported	Date and time the record was marked as exported.
prefix	A static alphanumeric purchase request number prefix.

This complex type supports the read method.

oaRatecard

Use this complex type to map job codes to hourly rates. oaRatecard has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
notes	Notes associated with the rate card.
active	A 1/0 field indicating whether this is an active rate card.
name	The name of the rate card.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaRateCardItem

Use this complex type to specify rate card item information. oaRateCardItem has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
rate_card_id	The ID of the rate card that it is associated with.
job_code_id	The ID of the associated job code.
currency	Currency for the money fields in the record.
updated	Time the record was last updated or modified.
rate	The hourly billing rate.
start	Start date of the rate for historical records.
end	End date of the rate for historical records.
current	A 1/0 field indicating if the record is the current rate.

This complex type supports the following methods: read, add, modify, and upsert.

oaReimbursement

Use this complex type to specify reimbursement information. oaReimbursement has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
envelopeid	The associated envelope the reimbursement is applied to.
date	The date of the reimbursement.
total	The reimbursement total. Dated by the date field.
updated	Time the record was last updated or modified.
currency	Currency for the money fields in the record.
envelope_number	The number of the associated envelope the reimbursement is applied to.
externalid	If the record was imported from an external system, you store the unique external record ID here.
notes	Notes associated with the reimbursement.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaRepeat

Use this complex type to specify repeating event information. oaRepeat has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
frequency	The repeating interval of the event: D – daily, W – weekly, M – monthly, Y – yearly.
every	The spacing between each repeating event.
end	End date of the event.
occur_number	Number of occurrences.
how_end	How does this event end: D – date or O - occurrence
exclude_dow	When frequency is in days, which days of the week (e.g. Mon, Tue, etc.) to exclude. This is a comma delimited list with 0 being Mon.
created	Time the record was created.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaReport

Use this complex type to hold saved report definitions and settings. oaReport has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
userid	The ID of the user who created the report. This is 0 for standard reports.
name	The name of the report.
type	The type of report: S = saved reports and T = sTandard reports.
thin_client_context	A 1/0 field indicating that this report can be requested via thin clients.
date_created	The date and time the report was created. This may or may not be the same as the created column. For example, reports created before 2010-01-11 and reports copied from other accounts will have different values.
email_report	A 1/0 field. 1 = report executes and sends an email with a pdf attachment to the session user.
relatedid	Related ID for attributes type. Report = ID of saved report, Timesheet = ID of timesheet, and Envelope = ID of related expense for expense report.

This complex type supports the read method.

oaRequest_item

Use this complex type to specify a request item, a single entry in a purchase request. oaRequest_item has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
vendor_quote_number	The vendor's quote number.
userid	The ID of the requester. This is always the same as the purchaserequest requester (user_id).
date	The date of the request_item. The same as the purchaserequest date.
manufacturerid	The ID of the associated manufacturer.
purchaserequestid	The ID of the associated purchaserequest.
um	The unit of measure for the product, i.e., EA.
request_reference_number	Unique reference number within purchase request.
attachmentid	If non-zero, the attachment record associated with this request_item.
currency	The currency used for this request item.
quantity_fulfilled	The quantity that has been fulfilled.
productid	The ID of the associated product.
date_fulfilled	The date on which all of the quantity was fulfilled.
purchase_itemid	The ID of the associated purchase_item.
allow_vendor_substitution	A 1/0 field indicating whether the vendor may be substituted.
manufacturer_part	The manufacturer's part number, SKU or other unique ID for this product.
purchaseorderid	The ID of the associated purchase order.
cost	The cost per unit of measure at which the product is being requested. 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
name	The request item name.
quantity	The quantity of product_id for this request item.
projectid	The ID of the associated project. This is always the same as the purchase request project_id.
total	The total value of the request item. Dated by the date field.
vendorid	The ID of the associated vendor.
vendor_sku	The vendor's sku for this product.
notes	Notes associated with this request item.
customerid	The ID of the associated customer. This is always the same as the purchase request customer_id.
exported	Date and time the record was marked as exported.

This complex type supports the read method.

oaResourceprofile

Use this complex type to specify items the make up a resource profile. oaResourceprofile has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the user for which this resourceprofile describes.
resourceprofile_typeid	The ID of the resourceprofile_type.
name	The resourceprofile name. Stub.
updated	Time the record was last updated or modified.
attributeid	The ID of the optional resourceprofile attribute.
comment	Additional comment describing this resourceprofile.
externalid	If the record was imported from an external system you store the unique external record ID here.
type	The resourceprofile type. The entity on which this resourceprofile is based. Skill Education Location Jobrole Industry Customprofile_1..20

This complex type supports the following methods: read, add, modify, and upsert.

oaResourceprofile_type

Use this complex type to specify information about a resource profile type. oaResourceprofile_type has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
active	A 1/0 field indicating whether this is active.
related_table	The name of the table related with this table.
name	The resourceprofile_type name. This shows up on all the resourceprofile_type pop-up windows in the application.
updated	Time the record was last updated or modified.

Field Name	Description
relatedid	The ID of the related item in the related table.
externalid	If record was imported from an external system, store the unique external record ID here.
type	The resourceprofile type. The entity on which this resourceprofile is based. Skill Education Location Jobrole Industry Customprofile_1..20

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaRevenueContainer

Use this complex type to specify information about the revenue_container header table. oaRevenueContainer has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
number	The revenue_container number that increments by 1.
date	The date of the revenue_container.
balancing_type	A one-character key indicating the type of balancing for this revenue_container. Note that All revenue_containers for a project have the same balancing_type: A - Agreement C - CustomerPO P - Project X - Agreement and CustomerPO
total_recognized	The revenue_container recognized total. Dated by the date field.
currency	The currency of this revenue_container.
date_approved	The date the invoice was approved.
updated	Time the record was last updated or modified.
date_submitted	The date the invoice was submitted.
approval_status	The approval status of the invoice. Only used if invoice approvals are used O - Open S - Submitted A - Approved R - Rejected
total_deferred	The revenue_container deferred total. Dated by the date field.
name	The name of the revenue_container (Prefix + number).
acct_date	The accounting period date of the revenue_container.
total_accrued	The revenue_container accrued total. Dated by the date field.

Field Name	Description
projectid	The ID of the associated project.
externalid	If the record was imported from an external system, you store the unique external record ID here.
total_posted	The revenue_container posted total. Dated by the date field.
created	Time the record was created.
notes	Notes to print on the revenue_container.
total_invoiced	The revenue_container invoice total. Dated by the date field.
customerid	The ID of the associated customer.
exported	Date and time the record was marked as exported.
prefix	A static alphanumeric revenue_container number prefix.

This complex type supports the read method.

oaRevenue_recognition_rule

Use this complex type to specify revenue recognition rules. oaRevenue_recognition_rule has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
user_filter	CSV list of users to limit the rule to.
purchase_how	How purchases should be recognized: M - mark up/down on billed purchases. B - billed purchases.
percent_complete	The calculated percent complete value if a type P transaction.
percent	The percentage value for a fixed fee percent trigger.
end_milestone	ID of the ending milestone (project_task).
recognition_type	What we are recognizing: R - revenue C - cost O - other
marked_as_ready	Trigger recognition when a task (ID in phase) is marked as ready to recognize.
break_by_user	Break out the transactions by user. Currently only implemented for the incurred rules.

Field Name	Description
percent_how	How percent complete should be calculated: A - % complete of planned hours for the project. B - % complete of planned hours for a phase. C - Approved hours versus planned hours for the project. D - Approved hours versus planned hours for a phase. E - Approved hours versus budget hours for the project.
timetype_filter	CSV list of timetypes to limit the rule to.
expense_how	How expenses should be recognized: M - mark up/down on billed expenses. B - billed expenses. I - incurred expenses.
active	A 1/0 field indicating whether this is an active rule.
name	Name of the rule.
categoryid	The ID of the associated category.
start_milestone	ID of the starting milestone (project_task).
end_date	End date of the rule.
customerid	The ID of the associated customer.
agreementid	ID of the associated agreement.
type	The type of the rule: A - as billed rule P - percent of time complete rule E - expense incurred rule F - fixed amount rule U - purchase item rule I - incurred versus forecast rule T - generated from a time project billing rule
asb_exclude_slip_type	CSV list of the slip types to exclude from the as billed rule.
customerpo_id	ID of the associated customerpo.
percent_trigger	If the fixed fee is triggered by a percent complete, this holds how it is triggered: A - % complete of planned hours for the project. B - % complete of planned hours for a phase or task (the task ID is held in the phase field).
item_filter	CSV list of items to limit the rule to.
project_task_filter	CSV list of tasks to limit the rule to.
product_filter	CSV list of products to limit the rule to.
slip_stage_filter	CSV list of slip_stage ID to limit a type A rule to.
repeatid	The ID of the associated repeating event.
currency	Currency for the money fields in the record.

Field Name	Description
phase	ID of the phase if percent_how is B or D. ID of the phase/task if this is a marked_as_ready or percent_trigger rule.
acct_code	Optional accounting system code for integration with external accounting systems.
start_date	Start date of the rule.
projectid	The ID of the associated project.
amount	The amount. If we have multiple amounts, the values are held in the revenue_recognition_rule_amount table.
extra_data	Holds extra data fields associated with the rule.
acct_date	The accounting period date to assign to the transaction.
acct_date_how	The accounting period date of the transaction is determined by: N - none, clear the value E - the entity (no change) C - container of the entity if available (i.e., timesheet, envelope) M - set by the specified accounting date P - set by the specified accounting period
accounting_period_id	The ID of the associated accounting period.
notes	Notes associated with this revenue recognition rule.
cost_centerid	The ID of the associated cost center.
asb_which_slips	Which slips should be considered for the as billed rule: A - all slips I - slips on invoices P - slips on approved invoices
project_billing_rule_filter	CSV list of project billing rule id's to limit a type T rule to.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.
assigned_user	The user to assign to fixed fee recognition.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenue_recognition_rule_amount

Use this complex type to specify multiple amounts for a recognition rule. oaRevenue_recognition_rule_amount has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
revenue_recognition_rule_id	The ID of the associated rule.

Field Name	Description
customerpo_id	The ID of the associated customerpo.
recognition_type	Recognition type: R - revenue C - cost O - other
updated	Time the record was last updated or modified.
currency	Currency for the money fields in the record.
category_id	The ID of the associated category.
amount	The amount.
acct_code	Optional accounting system code for integration with external accounting systems.
agreement_id	The ID of the associated agreement.
cost_center_id	The ID of the associated category.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id.
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenue_recognition_transaction

Use this complex type to specify revenue recognition transactions. This is a record of a single transaction created when revenue recognition was run for a particular project. oaRevenue_recognition_transaction has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
percent_complete	The calculated percent complete value if it is a type P transaction.
revenue_recognition_ruleid	The ID of the associated rule.
userid	The ID of the associated user.
date	The date of the transaction.
taskid	The ID of the associated task.
customerpo_id	The ID of the associated customerpo.

Field Name	Description
recognition_type	Recognition type: R - revenue C - cost O - other
updated	Time the record was last updated or modified.
slipid	The ID of the associated slip.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
ticketid	The ID of the associated ticket.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
total	The amount of the transaction. Dated by the date field.
categoryid	The ID of the associated category.
notes	Notes associated with this revenue recognition transaction.
acct_code	Optional accounting system code for integration with external accounting systems.
type	The type of the transaction. Matches the type field in revenue_recognition_rule.
acct_date	The accounting period date of the transaction.
cost_center_id	The ID of the associated cost center.
agreementid	The ID of the associated agreement.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
project_billing_ruleid	The ID of the associated project billing rule.
job_codeid	The ID of the associated job code.
rate	The hourly rate for a T type. Dated by the date field.
decimal_hours	The number of decimal hours.
hour	The number of hours for a T type.
minute	The number of minutes for a T type.
revenue_containerid	The ID of the associated revenue_container once posted.
revenue_stageid	The ID of the associated revenue stage.
originatingid	The ID of the originating revenue_recognition_transaction for this revenue_recognition_transaction.
portfolio_projectid	The ID of the associated portfolio project

Field Name	Description
offsetsid	The ID of the revenue_recognition_transaction which this revenue_recognition_transaction offsets.
is_from_open_stage	A 1/0 field indicating that the revenue recognition transaction was added to the revenue container from the virtual open stage, otherwise the transaction was added through revenue container revenue_recognition_transaction generation logic. If revenue_container_id is zero, revenue_stage_id should be 0 and is_from_open_stage should be 0.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenueStage

Use this complex type to specify revenue recognition transaction stage information. Index the attributes and use them to filter revenue recognition transactions. oaRevenueStage has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The name of the revenue stage.
revenue_stage_type	A one-character key indicating the type of revenue for this revenue_stage: D - Deferral A - Accrual F - Final
created	Time the record was created.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaSchedulebyday

Use this complex type to retrieve the day-by-day representation of users' work schedules. oaSchedulebyday has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
date	The date.
created	Time the record was created.
hours	The number of schedule hours on this date for this user, including exceptions.
user_id	The id of the associated user.
base_hours	The number of base hours on this date for this user.

Field Name	Description
target_hours	The number of target hours for this user on this date. Target_utilization.percentage * hours.
target_base_hours	The number of target base hours for this user on this date Target_utilization.percentage * base_hours.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaScheduleexception

Use this complex type to describe changes to the default work schedule for a company or user. oaScheduleexception has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
workhours	The number of hours per day during this daterange. This overrides any workhours on each day of either the account schedule or the account/user schedule.
userid	The ID of the user of this is an exception to the user's work schedule. 0 if this is an exception to an account work schedule.
name	The exception name and/or description, e.g. New Years Day.
exception_type	The type of exception. R - Date range of the exception.
startdate	The start date for the exception.
enddate	The end date for the exception.
updated	Time the record was last updated or modified.
workscheduleid	The ID of the corresponding work schedule.
timetypeid	The ID of the associated time type.
schedule_request_itemid	The ID of the schedule change item from a schedule request.

This complex type supports the read method.

oaSchedulerequest

Use this complex type to specify schedule request details. oaSchedulerequest has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.

Field Name	Description
created	Time the record was created.
number	The schedule request number that increments by 1.
userid	The ID of the user creating the schedule request.
startdate	The start date of the schedule request.
date	The date of the schedule request creation.
attachmentid	If non-zero, the attachment record associated with this schedule request.
enddate	The end date of the schedule request.
approval_status	The approval status of the schedule request: O - open P - pending approval A - approved R - rejected
updated	Time the record was last updated or modified.
date_approved	The date the schedule request was approved.
date_submitted	The date the schedule request was submitted.
customerid	The ID of the associated customer.
timetype	The time type of this schedule request: R - regular time or P - personal time.
timetypeid	The ID of the associated time type.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
total	The amount of the transaction. Dated by the date field.
categoryid	The ID of the associated category.
notes	Notes to print on the schedule request.
externalid	If the record was imported from an external system, you store the unique external record ID here.
description	The description or purpose for the schedule request.
prefix	A static alphanumeric schedule request number prefix.
name	The name of the schedule request (Prefix + number).

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaSchedulerequest_item

Use this complex type to specify information for multiple schedule request items. oaSchedulerequest_item has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.

Field Name	Description
hours	The number of hours represented by this schedule request item.
date	The date of the schedule request item.
userid	The ID of the associated user.
timetypeid	The ID of the associated time type.
name	The schedule request item name. It is the same as the schedule request description.
request_reference_number	Unique reference number within schedule request.
schedule_requestid	The ID of the associated schedule request.
updated	Time the record was last updated or modified.
customerid	The ID of the associated customer.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
categoryid	The ID of the associated category.
externalid	If the record was imported from an external system, you store the unique external record ID here.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaSlip

Use this complex type to specify slip information. A slip is an individual timebill or an individual charge to a customer. Multiple slips are aggregated into an invoice. oaSlip has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
hour	The number of hours for a T slip.
date	The date of the billing slip.
unitm	The unit of measure for an E or P slip or the rate description for an O slip.
rate	The hourly rate for a T slip. Dated by the date field.
slip_stageid	The ID of the associated slip stage.
project_billing_ruleid	The ID of the associated project billing rule.
cost	The cost per unit of measure for an E or P slip, the billing rate for an O slip, or the fixed price for a F slip. Dated by the date field.
tax_location_name	The name of the tax location
sold_to_contactid	The ID of the contact sold to.
description	The description of the billing slip.
total	The total value of the slip. Dated by the date field.

Field Name	Description
categoryid	The ID of the associated category. When set, the slip is based on this category.
timer_start	The starting time of the timer.
minute	The number of minutes for a T slip.
customerid	The ID of the associated customer.
type	The type of the slip: T - hourly rate slip E - expense slip F - flat price slip O - other time slip M - incomplete slip P - product slip
agreementid	The ID of the associated agreement.
total_tax	The total tax paid. Dated by the date field.
customerpoid	The ID of the associated customerpo.
userid	The ID of the associated user.
invoioceid	The ID of the associated invoice once billed.
currency	Currency for the money fields in the record
city	The slip city or location.
decimal_hours	The number of decimal hours for a T slip.
cost_centerid	The ID of the associated cost center.
payment_typeid	The ID of the associated payment type.
total_with_tax	A 1/0 field indicating whether the cost includes the tax.
shipping_contactid	The ID of the associated shipping contact.
itemid	The ID of the associated item. If this is set, the slip is based on this item. Determine the subtype using the associated item type.
timetypeid	The ID of the associated time type.
quantity	The quantity for an E, O, or P slip.
billing_contactid	The ID of the associated billing contact.
projectid	The ID of the associated project.
projecttaskid	The ID of the task within the associated project.
productid	The ID of the associated product.
acct_date	The accounting period date of the slip.
notes	Notes associated with the slip.
projecttask_type_id	The ID of the projecttask_type of the associated projecttask.
job_code_id	The ID of the associated job code.
payroll_type_id	The ID of the associated payroll type.
ref_slipid	For credit/rebill, ID of the original slip id.
portfolio_projectid	The ID of the associated portfolio project.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.

Field Name	Description
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
gl_code	The fixed code 1234455454.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: If not all slips are returned from an API request, determine whether one or both of the following OpenAir internal switches are enabled:

- API should convert charges money fields to invoice currency. Only invoiced slips are returned.
- API should filter out charges associated with charge stages marked to be excluded from invoicing.

To enable or disable them, speak with your Professional Services Consultant or create a support ticket. See Troubleshooting for instructions on creating a support ticket.

oaSlipProjection

Use the oaSlipProjection complex type to hold slips created from a projected billing run. This complex type includes many of the oaSlip fields with additional fields. The oaSlipProjection has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
hour	The number of hours for a T slip.
date	The date of the billing slip.
unitm	The unit of measure for an E or P slip or the rate description for an O slip.
rate	The hourly rate for a T slip. Dated by the date field.
slip_stageid	The ID of the associated slip stage.
project_billing_ruleid	The ID of the associated project billing rule.
cost	The cost per unit of measure for an E or P slip, the billing rate for an O slip, or the fixed price for a F slip. Dated by the date field.
sold_to_contactid	The ID of the contact sold to.
description	The description of the billing slip.
total	The total value of the slip. Dated by the date field.
categoryid	The ID of the associated category. When set, the slip is based on this category.
timer_start	The starting time of the timer.

Field Name	Description
minute	The number of minutes for a T slip.
customerid	The ID of the associated customer.
type	The type of the slip: T - hourly rate slip E - expense slip F - flat price slip O - other time slip M - incomplete slip P - product slip
agreementid	The ID of the associated agreement.
customerpoid	The ID of the associated customerpo.
userid	The ID of the associated user.
invoioceid	The ID of the associated invoice once billed.
currency	Currency for the money fields in the record
city	The slip city or location.
decimal_hours	The number of decimal hours for a T slip.
payment_typeid	The ID of the associated payment type.
shipping_contactid	The ID of the associated shipping contact.
itemid	The ID of the associated item. If this is set, the slip is based on this item. Determine the subtype using the associated item type.
timetypeid	The ID of the associated time type.
quantity	The quantity for an E, O, or P slip.
billing_contactid	The ID of the associated billing contact.
projectid	The ID of the associated project.
projecttaskid	The ID of the task within the associated project.
productid	The ID of the associated product.
notes	Notes associated with the slip.
slip_projection_type	The type of the slip projection: X - slip projection generated from billing rule, B - Time from potentially billable transaction which did not match any billing rule, N - Time from transaction with non-billable project-task, P - Time from transaction matching a billing rule but is Partially over cap, S - Time from transaction matching a billing rule but is completely over cap and rule indicates to Stop if capped C - Time from transaction matching a billing rule but is completely over cap and no more rules match.
booking_typeid	ID of the booking type if this was generated from bookings.
transactionid	For internal use only.

This complex type supports the read method.

oaSlipstage

Use this complex type to specify the various stages a slip can be in. oaSlipstage has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
exclude_from_invoicing	Exclude slips of this stage from invoicing.
notes	Notes associated with this slip stage.
name	The name of the stage.
updated	Time the record was last updated or modified.
position	The position of the stage.
enable_slip_tab	Display slips of this stage in a separate tab.

This complex type supports the following methods: read, add, modify, and upsert.

oaSwitch

Use this complex type to specify information about company and user switches. oaSwitch has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
name	The name of the switch setting.
setting	The contents of the switch setting. A zero length field is considered 'undef'.

Refer to oaCompany and oaUser for more information.

oaTagGroup

Use this complex type to specify user entity tags for users, customers, or projects. oaTagGroup has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
active	A 1/0 field indicating whether the record is active.
updated	Time the record was last updated or modified.
name	Name of the tag group.
entity_type	The tag group type: U - user, C - customer, or P - project.
searchable	A 1/0 field indicating whether this tag group is searchable. Used only for tag group type = U.

This complex type supports the read method.

oaTagGroupAttribute

Use this complex type to specify attributes associated with user entity tags for users, customers, or projects. oaTagGroupAttribute has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
active	A 1/0 field indicating whether the record is active.
updated	Time the record was last updated or modified.
name	Name of the tag group attribute.
tag_group_id	The ID of the tag group this attribute is in.
attributes	A collection of additional attributes for this complex type.

This complex type supports the read method.

oaTargetUtilization

Use this complex type to specify target utilization values for a user. oaTargetUtilization has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
user_id	The ID of the associated user.
start_date	The start date for the target utilization.
end_date	The end date for the target utilization. This field is automatically determined based on the next subsequently later start date row for the user. This field can be 0000-00-00 for one row which represents the unbounded value.
percentage	Target utilization for this user as a percentage. For example, 75.30.
dirty	A 2/1/0 field: 0 - Clean 1 - Dirty 2 - Inprogress

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaTask

Use this complex type to specify task information. oaTask has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
projecttask_typeid	The ID of the projecttask_type of the associated project_task.
userid	The ID of the associated user.
date	The date of the task.
decimal_hours	The number of decimal hours for the task.
cost_centerid	The ID of the associated cost center.
slipid	The ID of the associated slip if this task was billed.
hours	The number of hours for the task.
timetypeid	The ID of the associated time type.
minutes	The number of minutes for the task.
projectid	The ID of the associated project.
description	Description of the task.
categoryid	The ID of the associated category.
projecttaskid	The ID of the task within the associated project.
timesheetid	The ID of the associated timesheet.
notes	Notes associated with this task.
customerid	The ID of the associated customer.
job_codeid	The ID of the associated job code.
payroll_typeid	The ID of the associated payroll type.
loaded_cost	Loaded cost for the associated resource, using the forex rate on the task date.
loaded_cost_2	User's second level loaded cost, using the forex rate on the task date.
loaded_cost_3	User's third level loaded cost, using the forex rate on the task date.
project_loaded_cost	User's project cost override in project currency.
project_loaded_cost_2	User's project second cost in project currency.
project_loaded_cost_3	User's project third cost in project currency.
acct_date	The accounting period date of the task.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
thin_client_id	Used by thin clients to reconcile imported records.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaTaskTimecard

Use this complex type to specify tasks associated with timecards. oaTaskTimecard has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
projecttask_typeid	The ID of the project task type.
project_phaseid	The ID of the project phase.
userid	The ID of the associated user.
date	The date of the task timecard.
decimal_hours	The number of decimal hours for the task timecard.
cost_centerid	The ID of the associated cost center.
slipid	The ID of the associated slip.
hours	The number of hours for the task timecard.
timetypeid	The ID of the associated time type.
minutes	The number of minutes for the task timecard.
projectid	The ID of the associated project.
description	The description of the task timecard.
categoryid	The ID of the associated category.
projecttaskid	The ID of the task within the associated project.
timesheetid	The ID of the associated timesheet.
notes	Notes associated with this task timecard.
time_cardid	The ID of the associated timecard.
customerid	The ID of the associated customer.
payroll_typeid	The ID of the associated payroll type.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.

This complex type supports the read method.

oaTaxLocation

Use this complex type to specify tax location information. oaTaxLocation has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
hst_rate	The HST rate. This is used instead of GST and PST in some locations.
federal_rate	The federal tax rate.
name	The name for the estimate adjustment.
updated	Time the record was last updated or modified.
acct_code_federal	GL accounting code for the federal entries.
tax_method	The tax method: G - GST and PST H - HST F - Federal and State
state_rate	The state tax rate.
acct_code_pst	GL accounting code for the PST entries.
acct_code_state	GL accounting code for the state entries.
active	A 1/0 field specifying if the location is active.
acct_code_gst	GL accounting code for the GST entries.
pst_rate	The PST rate.
acct_code_hst	GL accounting code for the HST entries.
notes	Notes associated with this tax location.
gst_rate	The GST rate.

This complex type supports the following methods: read, add, modify, and upsert.

oaTaxRate

Use this complex type to specify tax rate information. oaTaxRate has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
pst	The PST tax. Dated by the date field.
date	The date (used for currency conversions).
notes	Notes associated with this tax rate.
updated	Time the record was last updated or modified.

Field Name	Description
federal	The federal tax. Dated by the date field.
tax_locationid	The ID of the associated tax location.
state	The state tax. Dated by the date field.
currency	Currency for the money fields in the record.
hst	The HST tax. Dated by the date field.
slipid	The ID of the associated slip.
ticketid	The ID of the associated ticket.
gst	The GST tax. Dated by the date field.
purchase_itemid	The ID of the associated purchase order item.

This complex type supports the following methods: read, add, modify, and upsert.

oaTerm

Use this complex type to specify term information. oaTerm has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
name	The name for the term.
display	Display the term as.

This complex type supports the read method.

oaTicket

Use this complex type to specify ticket information. oaTicket has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
date	The date of the ticket.
unitm	The unit of measure.
reference_number	Unique reference number within envelope. Used to cross-reference digital receipts with paper receipts.
currency_total_tax_paid	The tax paid in the foreign currency if this is a foreign currency receipt.
tax_rateid	The ID of the associated tax rate.
currency_rate	The conversion rate if this is a foreign currency receipt.
total_no_tax	The total paid before tax added. Dated by the date field.

Field Name	Description
project_taskid	The ID of the associated project task.
cost	The cost per unit of measure. Dated by the date field.
tax_location_name	The name of the tax location.
non_billable	If set to 1, this is not billable.
description	The description of the ticket.
total	The total value of the ticket. Dated by the date field.
categoryid	The ID of the associated category.
customerid	The ID of the associated customer.
paymethod	Payment method now comes from payment_type table. Keep for backwards compatibility.
userid	The ID of the associated user.
status	The status of the ticket: R - reimbursable N - non-reimbursable
currency	Currency for the money fields in the record.
city	The ticket city or location.
cost_centerid	The ID of the associated cost center.
slipid	The ID of the associated slip.
currency_cost	The cost per unit of measure in the foreign currency if this is a foreign currency receipt.
payment_typeid	The ID into the payment type field for the payment method.
currency_symbol	The currency for foreign currency receipts.
tax_location_id	The ID of the associated tax location.
itemid	The ID of the associated item.
envelopeid	The ID of the associated envelope.
quantity	The quantity.
projectid	The ID of the associated project.
total_tax_paid	The tax paid. Dated by the date field.
vendorid	The ID of the associated vendor.
missing_receipt	If set to 1, the paper receipt is missing for this ticket.
notes	Notes associated with the ticket.
attachmentid	If non-zero, the attachment record associated with this ticket.
currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance as defined in database data definitions.
externalid	If the record was imported from an external system, store the unique external record ID here.
acct_date	The accounting period date of the ticket.
projecttask_typeid	The ID of the associated projecttask_type. Only if project_task_id switch is on.
thin_client_id	Used by thin clients to reconcile imported records.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: There is an OpenAir internal switch that prohibits you from editing the following fields created through the American Express receipt import wizard: date, quantity, cost, currency, payment_typeid, and total. However, in the event that editing is necessary, you can request that the following switch be temporarily disabled: Do not allow editing of receipts with an American Express transaction number. To enable or disable the internal switch, speak with your Professional Services Consultant or create a support ticket. See Troubleshooting for instructions on creating a support ticket.

oaTimecard

Use this complex type to specify timecard information. oaTimecard has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
time_start	Time they started working.
hours	Hours worked.
notes	Notes associated with the timecard.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.
date	The date of the timecard.
break_end	Time they ended the break.
break_start	Time they started the break.
timesheetid	The ID of the associated timesheet.
time_end	Time they stopped working.

This complex type supports the read method.

oaTimesheet

Use this complex type to specify timesheet information. oaTimesheet has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.
status	The status of the timesheet: O - Open S - Submitted A - Approved R - Rejected X - Archived
default_payrolltypeid	The default payroll type ID this timesheet is associated with.
default_timetypeid	The default time type ID this timesheet is associated with.
name	The name of the timesheet.
default_customerid	The default customer ID this timesheet is associated with. All new task entries get this default value.
submitted	The date the timesheet was submitted.
total	The total number of hours in the timesheet.
default_categoryid	The default category ID this timesheet is associated with. All new task entries get this default value.
ends	The ending date of the timesheet.
starts	The starting date of the timesheet.
approved	The date the timesheet was approved.
notes	Notes related to this timesheet.
default_projectid	The default project ID this timesheet is associated with.
acct_date	The accounting period date of the task.
thin_client_id	Used by thin clients to reconcile imported records.
history	History of events that occurred to the TimeSheet.
approved_by	Empty value kept for backwards compatibility.
duration	The duration of the timesheet: W - Weekly D - Daily M - Monthly B - Bi-weekly S - Semi-monthly

Field Name	Description
default_projecttaskid	The default task id this timesheet is associated with. All new task entries get this default value.
default_per_row	Holds a data structure of per row defaults. The format is as follows: Multiple CSV rows with each row having the element name ('cp','category' etc.) as the first record and then the id values per row.

This complex type supports the following methods: read, add, modify, upsert, submit, and delete.

Note: Refer to the following notes regarding the Timesheet datatype:

- To be able to edit an approved or archived timesheet, the following internal switch must be enabled: API will allow editing of approved and archived Timesheets.
- If the following switches are enabled, timesheets cannot be edited:
  - Do not allow the owner to edit a submitted timesheet
  - Disable editing of exported timesheets

A user who attempts to modify another user's timesheet must have a full Account Administrator role. Refer to Add/Modify Errors for more information on error code 821 relating to Timesheets.

If you would like to determine whether any of these internal switches are enabled, speak with your Professional Services Consultant or create a support ticket. See Troubleshooting for instructions on how to create a support ticket.

oaTimetype

Use this complex type to specify information for time types such as regular time, overtime, sick time. oaTimetype has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
notes	Notes associated with this time type.
active	A 1/0 field indicating whether this is time type is active.
name	The name of the time type.
updated	Time the record was last updated or modified.
externalid	If the record was imported from an external system, you store the unique external record ID here.
payroll_code	The payroll code for this time type.

Field Name	Description
cost_centerid	The ID of the associated cost center.
code	Optional accounting system code for integration with external accounting systems.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaTodo

Use this complex type to specify information about something that needs to be done. oaTodo has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
priority	Todo priority (1 - 9).
contactid	The ID of the associated contact.
name	The name or description of the todo item.
updated	Time the record was last updated or modified.
due	Date and time the task is due.
userid	The ID of the associated user.
dealid	The ID of the associated deal.
status	Todo status: A - Active C - Completed D - Deferred N - Not Started W - Waiting
notes	Notes associated with the todo item.
customerid	The ID of the associated customer.
createdbyid	The ID of the user who created the todo item.
finished	Date and time the task was finished.
start	Date and time the task is to be started.

This complex type supports the read method.

oaUprate

Use this complex type to specify information about user and project rate combinations. oaUprate has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the associated user.
customerid	The ID of the associated customer.
notes	Notes associated with the user project rate (uprate).
updated	Time the record was last updated or modified.
duration	Billing rate: H - hourly D - Daily
projectid	The ID of the associated project.
categoryid	The ID of the associated category.
currency	Currency for the money fields in the record.
rate	The billing rate.
project_billing_ruleid	If project billing rules are used, this is the ID of the associated project billing rule.
job_codeid	The ID of the job code this rate is associated with. This is only used in the context of project billing rules.

This complex type supports the following methods: read, add, modify, and upsert.

oaUser

Use this complex type to specify user information. oaUser has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
project_access_nodes	Comma delimited list of hierarchy node IDs for project level access control.
addr_mobile	Mobile number.
addr_country	The country.
po_approver	The user_id of the purchase order approver if this is a single user approver process. This field is mutually exclusive with po_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
rate	The hourly billing rate.
br_approvalprocess	The approvalprocess_id of the deal_booking_request approval process. This field is mutually exclusive with br_approver.

Field Name	Description
password	The user's password.
te_approver	The user_ID of the expense report approver if this is a single approver process. This field is mutually exclusive with te_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
sr_approver	The user ID of the schedule request approver if this is a single approver process. This field is mutually exclusive with sr_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
departmentid	The ID of the associated department.
tb_filter_set	The ID of the optional filter set for the Invoices module.
addr_last	The user's last name.
name	The name used for display in lists. This is programmatically generated if not entered.
hierarchy_node_ids	The IDs of the associated hierarchy nodes.
po_approvalprocess	The approvalprocess_id of the purchase order approval process. This field is mutually exclusive with po_approver.
addr_fax	The user's fax number.
rm_filter_set	The ID of the optional filter set for the Resources module.
addr_city	The city.
hint	Password hint.
dr_approver	The user ID of the deal booking request approver if this is a single approver process. This field is mutually exclusive with dr_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
br_approver	The user ID of the booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
az_approvalprocess	The approvalprocess_id of the expense authorization approval process. This field is mutually exclusive with az_approver.
pm_filter_set	The ID of the optional filter set for the Projects module.
currency	The currency for money fields.
cost_centerid	The ID of the associated cost center.
addr_zip	The zip code.
locked	A 1/0 field indicating if this user is locked.
filterset_stamp	A unique string which changes when the primary filter set changes for the user.
addr_addr1	Address line one.
job_codeid	The ID of the current job code this user belongs to.
payroll_code	The payroll code for this user.

Field Name	Description
addr_middle	The user's middle name.
report_filter_set	The ID of the optional filter set for Reporting.
addr_addr2	Address line two.
km_filter_set	The ID of the optional filter set for the Workspaces module.
az_approver	The user ID of the expense authorization approver if this is a single approver process. This field is mutually exclusive with az_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
addr_addr4	Address line four.
role_id	The ID of the associated role.
dr_approvalprocess	The approvalprocess_id of the deal_booking_request approval process. This field is mutually exclusive with br_approver.
te_approvalprocess	The approvalprocess_id of the expense report approval process. This field is mutually exclusive with te_approver.
ta_approvalprocess	The approvalprocess_id of the timesheet approval process. This field is mutually exclusive with ta_approver.
filterset_ids	A comma separated list of filter set IDs this record should be part of.
addr_first	The user's first name.
addr_addr3	Address line three.
active	A 1/0 field indicating where this is designated as an active user.
externalid	If the record was imported from an external system, you store the unique external record ID here.
ta_approver	The user ID of the timesheet approver if this is a single approver process. This field is mutually exclusive with ta_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
addr_salutation	The user's salutation.
generic	A 1/0 field indicating whether this is a generic resource.
pr_approver	The user ID of the purchase request approver if this is a single user approver process. This field is mutually exclusive with pr_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
pb_approvalprocess	The approvalprocess_id of the proposals approval process. This field is mutually exclusive with pb_approver.
type	Legacy field.
workscheduleid	The ID of the associated user workschedule.
po_filter_set	The ID of the optional filter set for the Purchases module.
addr_state	The state.
primary_filter_set	The ID of the primary filter set for this user.
user_locationid	The location ID for this user.
addr_phone	The user's phone number.

Field Name	Description
account_workscheduleid	The ID of the associated user account workschedule.
om_filter_set	The ID of the optional filter set for the Opportunities module.
ssn	The users's social security number.
acct_code	Optional accounting system code for integration with external accounting systems.
ma_filter_set	The ID of the optional filter set for the My Account module.
te_filter_set	The ID of the optional filter set for the Expenses module.
sr_approvalprocess	The approvalprocess_id of the schedule_request approval process. This field is mutually exclusive with sr_approver.
addr_email	The user's email address.
nickname	The users nickname. This must be unique.
pb_approver	The user ID of the booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
logintime	The date and time of the user's last login.
pr_approvalprocess	The approvalprocess_id of the purchase request approval process. This field is mutually exclusive with pr_approver.
line_managerid	The ID of this user's line manager (will actually be another user_id).
week_starts	The day the week starts for this user: 0 - Monday 6 - Sunday
ta_filter_set	The ID of the optional filter set for the Timesheets module.
flags	A collection of oaSwitch values.
update_workschedule	A 1/0 field indicating an update to the user's workschedule.
is_user_schedule	A 1/0 field indicating whether the user should draw their workschedule from an account_workschedule or draw from a custom workschedule. 0 sets the user workschedule to the account workschedule specified in account_workscheduleid, 1 constructs a custom workschedule from the supplied workschedule_workdays and workschedule_workhours fields.
workschedule_workdays	A CSV list of workdays, with each value indicating a day in the schedule and values ranging from 0(Monday) to 6(Sunday). For example, “0,1,4” indicates that a user works on Monday, Tuesday and Friday.
workschedule_workhours	A CSV list of values for the user's default workhours and workhours for each day. At least one value for workschedule_workhours must be submitted, but a value for each day may be submitted as well. For example, if the user's workschedule_workdays is set to “0,1,4”, then submitting a value of only “8” for workschedule_workhours sets the user's default hours to 8 and each workday assumes this value as well. In addition, submitting a workschedule_workdays value of “8,1,2,3” sets the user's default workhours to 8, sets Monday to 1, Tuesday to 2, and Friday to 3.
update_tag	Set this field to 1 to enable automatic updating of user entity tags.

Field Name	Description
tag_start_date	Start date for the new tag. If left blank, the start date for the new tag will be set to the current date.
tag_end_date	End date for the new tag. If left blank, the end date for the new tag will be undefined and the new tag will assume default status for the user.
tag_group_id	The ID of the tag group for the new tag.
tag_group_attribute_id	The ID of the tag group attribute that is being assigned to the new tag.
update_cost	Set this field to 1 to enable automatic updating of user loaded cost.
cost_start_date	Start date for the new loaded cost. If left blank, the new cost will assume the current date as it's start date.
cost_end_date	End date for the new loaded cost. If left blank, the new cost will have no end date.
cost	New cost value.
cost_currency	Currency of the cost.
cost_lc_level	If multiple loaded cost levels are enabled, use this field to hold the level of the loaded cost.
timezone	The user's timezone.
book_assign_stamp	Internal hash key.
br_approver	The user ID of the booking request approver if this is a single user approver process. This field is mutually exclusive with br_approvalprocess. 1 - approver is the manager. 2 - approver is the manager's manager.
code	The acct_code.
external_id	The unique external record ID if the record was imported from an external system .
password_forced_change	A 1/0 field indicating whether the password must change at next login.

This complex type supports the following methods: read, createUser, modify, upsert, and delete. Also refer to oaSwitch.

- In order to return generic users in a ReadRequest, add a "generic" attribute to the request. See generic.
- For generic users, you can use both add and upsert: generic flag=1.
- To create OpenAir users, use the createUser method instead of the add method.

Set User Workschedule

Refer to oaUserWorkschedule to read user workschedule information.

To set the user workschedule while updating or creating users:

1. Set the update_workschedule field of the oaUser object to 1.
2. To set up a user-specified work schedule, set the is_user_schedule flag to 1.

- Populate the workschedule_workdays field with a CSV list of user workdays. The values in the list should be numbers ranging from 0 (Monday) to 6 (Sunday). For example, 0,1,4 would mean that the user works Monday, Tuesday, Friday, while populating the field with a value of just 0 would mean the user only works on Monday.
- Populate the workschedule_workhours field with a CSV list of hours to be worked each day. The first value corresponds to the Default value, while subsequent values correspond to the days specified in workschedule_workdays. Using the above example, 8,1,2,3 would set the default workhours value to 8, Monday to 1, Tuesday to 2 and Friday to 4.

  Note: If the internal switch "Enable distinct work hours per day on workschedule" is not set, the workschedule_workhours field should only contain one value, the default.

3. Set the is_user schedule flag to 0 to use the company work schedule specified in the account_workscheduleid field.

Update User Entity Tags Automatically

To automatically update a user's entity tags, set the following fields in the oaUser object:

1. update_tag: Set this field to 1 to enable automatic updating of user entity tags.
2. tag_start_date: Start date for the new tag. If left blank, the start date for the new tag will be set to the current date.
3. tag_end_date: End date for the new tag. If left blank, the end date for the new tag will be undefined and the new tag will assume default status for the user.
4. tag_group_id: ID of the tag group for the new tag.
5. tag_group_attribute_id: ID of the tag group attribute that is being assigned to the new tag.

Refer to the following example for initial imports:

If the user has no tags currently set and a modify is performed, the user will receive a new default tag with a start date of the current date and the supplied tag_group_id and tag_group_attribute_id.

1. Set update_tag=1. (This enables automatic updating of user entity tag.)
2. Set tag_start_date=blank. (This indicates that the start date should be the current date.)
3. Set tag_end_date=blank.
4. Set tag_group_id=ID of a valid tag group.
5. Set tag_group_attribute_id=ID of a valid tag group attribute.

Refer to the following example for subsequent imports:

On subsequent imports of user tag information, the existing tags are automatically adjusted to accommodate the new tag. The previously imported tag's end date is set to the day before the start date of the new tag, i.e., yesterday, and the tag loses its default status. The new tag assumes default status and has a start date of the current date, i.e., today. Using the above example, assume the following fields were set during a modify on a user object.

1. Set update_tag=1.
2. Set tag_start_date=blank.
3. Set tag_end_date=blank.

After this update, the previously imported tag will have it's end date set to the day before the start date of the new tag (yesterday) and will also lose it's default status. The new tag will assume default status and will have a start date of today.

Update User Loaded Costs Automatically

The way you automatically update user loaded costs is similar to updating user entity tags, although there are a few key differences.

- First, default costs cannot be set using this method. All costs loaded using this method are treated as historical cost records.
- Second, only costs with the same cost_lc_level are compared when determining which historical records should be altered. If no cost_lc_level is specified, an lc_level of 0 is assumed.

To automatically update user loaded costs, the following fields should be set:

1. update_cost: Set this field to 1 to enable automatic updating of user loaded cost.
2. cost_start_date: Start date for the new loaded cost. If left blank, the new cost will assume the current date as it's start date.
3. cost_end_date: End date for the new loaded cost. If left blank, the new cost will have no end date.
4. cost: New cost value.
5. cost_currency: Currency of the cost.
6. cost_lc_level: If multiple loaded costs are enabled, use this field to hold the level of the loaded cost.

oaUserWorkschedule

Use this complex type to retrieve information about user-specific and company-wide work schedules. oaUserWorkschedule has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
name	The company-wide schedule name for company schedules or user's first and last name for user schedules.
userid	ID of the user if this is a users work schedule. 0 - if there is a company work schedule.
use_this_schedule	Can be 0 or 1. If "1" and userid has a value, then this is a user schedule (with userid above) which overrides the company schedule. If "1" and userid is 0, then this is a company schedule. If "0" then the user (with userid above) is using the company schedule indicated by account_workscheduleid.
account_workscheduleid	The ID of the company workschedule to use when userid in not 0.
workdays	A seven-letter string indicating which days of the week are available for project work. (Monday is 0, Sunday is 6; 01234 = Mon. - Fri.; 0123456 = every day). Always begins with the letter "x" (So "Monday only" would be "x0").
workhours	The number of hours worked per day.
created	Time the record was created
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaVendor

Use this complex type to specify vendor information. oaVendor has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
addr_state	The state.
addr_mobile	The mobile phone number.
addr_country	The country.
terms	Standard payment terms for the vendor.
addr_phone	The phone number.
addr_addr4	Address line four.
purchaseorder_text	Text to display on every purchase order.
currency	Currency for the money fields in the record. Also the default currency when a purchase order is created.
web	Vendor's Web address.
addr_zip	The zip code.

Field Name	Description
addr_first	The vendor's first name.
code	Optional accounting system code for integration with external accounting systems.
addr_email	The vendor's email address.
addr_addr3	Address line three.
attention	To whom purchase orders should be sent.
addr_addr1	Address line one.
addr_last	The vendor's last name.
name	Vendor name. Displays on all the vendor pop-up windows in the application.
active	A 1/0 field indicating where this is designated as an active vendor 1/0.
purchaseprder_email_text	Extra text to include in emails announcing purchase orders.
externalid	If record is imported from an external system, store external record ID here.
tax_locationid	The ID of the associated tax location.
addr_middle	The vendor's middle name.
addr_fax	Fax number.
addr_salutation	The vendor's salutation.
addr_city	The city.
addr_addr2	Address line two.
notes	Notes associated with this vendor.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaViewfilter

Use this complex type to filter lists or calendars. oaViewfilter has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
userid	The user who created this filter.
name	The internal name of the list or calendar this filter is applied to.
label	The name given to this filter. It appears in the Filter: drop-down list.
action	The filter action.
fields	Comma delimited list of fields.
match_all	A 1/0 field. 1 = if all rules met. 0 = if rules must be met.

This complex type supports the read method.

oaViewfilterrule

Use this complex type to specify the individual rules for a particular viewfilter. oaViewfilter has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
updated	Time the record was last updated or modified.
viewfilterid	The viewfilter to which this rule belongs.
field	The field or column to be compared.
type	The underlying type of the field or column to be compared: C = character string, N = number, D = date, B = Yes/No, P1 = picker_button, P2 = pop-up menu.
condition	One of the following conditions: ct = contains, nc = does not contain, eq = is equal to, ne = is not equal to, bw = begins with, ew = ends with, gt = is greater than, ge = is greater than or equal to, lt = is less than, le = is less than or equal to, in = in the set of.
value	The value the field is compared to.
required	A 1/0 field. 1 = if this condition must be met. 0 = if this is one of many that will satisfy this viewfilter. (If 1, this condition is ANDed with the others. If 0, this condition is ORed with the others.)

This complex type supports the read method.

oaWorkspacelink

Use this complex type to specify workspace associations with other records. oaWorkspacelink has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
recordid	The table ID the workspace is associated with.
url	The URL of external link.
external	A 1/0 field indicating if the record is an external link.
updated	Time the record was last updated or modified.
workspaceid	The ID of the associated workspace.

This complex type supports the following methods: read, add, modify, and upsert.

oaWorkspaceuser

Use this complex type to specify workspace user permission information. oaWorkspaceuser has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
userid	The ID of the associated user.
access	The access permissions for the user: R - Read-only W - Read/write A - Administrator
workspaceid	The ID of the associated workspace.
id	Unique ID. Automatically assigned by the system.
attributes	A collection of additional attributes for this complex type.

This complex type supports the following methods: read, add, modify, and upsert.

Chapter 7 Setting Application Switches Via the API

Certain company and user-level switches can be set via the API. Switches are settings that customize the application. They are not settings that affect actual record data.

Switches are set using the oaSwitch object. Currently, only the Company and User objects have a flags collection that holds individual oaSwitch objects. Switches set at the company level will affect an entire account; switches set at the user level will affect only a particular user in an account.

To obtain a list of switches supported by the system, please contact the OpenAir Support Department and open a support ticket. See Troubleshooting.

Chapter 8 Code Examples

Code examples are provided for login functions and read functions.

Login Functions

This section walks through a sample Java client application. Web service client access helper classes and stubs for this example were generated using the Apache Axis WSDL2Java tool. For more information on this tool, see Getting Started. The example demonstrates the following functions:

1. Login to the OpenAir Web Services using user-supplied credentials.
2. Create multiple new users in the logged-in user's account with user-supplied information.
3. Logout of the OpenAir web service.

Login Code Example

```
import java.rmi.RemoteException;
import javax.xml.soap.SOAPElement;
import org.apache.axis.message.SOAPHeaderElement;
import java.io.*;

class Program
{
      // Instance of our OpenAir web service proxy object
      private static OAirServiceSoapBindingStub m_svc;

      // Name of the company any new users will be added to
      private static String m_strCompany;

      // Prompts the user for input from the console
      private static String GetUserInput(String prompt)
      {
            {
                  BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
                  return reader.readLine();
            }
            catch (java.io.IOException e)
```

```
      {
              return null;
      }
}


// Logs into the OpenAir web service using the supplied login credentials.
// Returns true if successful, false if not
private static boolean Login() throws javax.xml.soap.SOAPException, javax.xml.rpc.ServiceException
{
      // setup our login information
      LoginParams lp = new LoginParams();
      lp.setApi_key("************");
      lp.setApi_namespace("company_namespace");
      lp.setUser( GetUserInput("Enter username: ") );
      lp.setPassword( GetUserInput("Enter password: ") );
      lp.setCompany( GetUserInput("Enter company: ") );
      m_strCompany = lp.getCompany();

      try
      {
           // get an instance of our service and login
           OAirServiceHandlerServiceLocator locator = new OAirServiceHandlerServiceLocator();
           m_svc = (OAirServiceSoapBindingStub)locator.getOAirService();

           LoginResult loginResult = m_svc.login(lp);
           System.out.println("Logged in, session ID = " + loginResult.getSessionId()+"\n");

           // set our session header to contain the session ID so we can perform further operations
           SOAPHeaderElement header = new SOAPHeaderElement("http://www.openair.com/OAirService",
           "SessionHeader");
           SOAPElement node = header.addChildElement("sessionId");
           node.addTextNode(loginResult.getSessionId());
           m_svc.setHeader(header);
      }
      catch (java.rmi.RemoteException e)
      {
           // catch any login problems and return
           System.out.println(e.toString());
           return false;
      }
      return true;
}
```

```
// Create a new user using user-supplied information
private static void CreateUser()
{
      System.out.println("----------------------------");
      System.out.println("Enter new user information\n");

      // create the company object that the new user will be associated with
      OaCompany company = new OaCompany();
      company.setNickname(m_strCompany);

      // get the new user information
      OaUser user = new OaUser();
      user.setNickname( GetUserInput("Enter username: ") );
      user.setRole_id( GetUserInput("Enter role ID: ") );
      user.setAddr_first( GetUserInput("Enter first name: ") );
      user.setAddr_last( GetUserInput("Enter last name: ") );
      user.setAddr_email( GetUserInput("Enter email: ") );
      user.setPassword( GetUserInput("Enter password: ") );
      try
      {
          // add the user and output any errors encountered
          UpdateResult result = m_svc.createUser(user, company);
          if (result.getErrors() != null)
          {
              for (OaBase base : result.getErrors())
              {
                  OaError err = (OaError)base;
                  System.out.println("Error: " + err.getCode() + "\t" +
                  err.getComment() + "\t" + err.getText());
              }
          }
          if (result.getStatus() == "A")
              System.out.println("User successfully added");
      }
      catch (Exception e)
      {
          System.out.println("Error while adding user:\n"+e.toString());
      }
}

// Application entry point
public static void main(String[] args)
{
      try
      {
          // Login to the OpenAir web service and add users
          if (Login())
          {
```

```
            {
                do
                {
                    CreateUser();
                } while (GetUserInput("\nAdd another (y/n)?
                ").toLowerCase().startsWith("y"));

                m_svc.logout();
            }
        }
        catch (Exception e)
        {
            System.out.println(e.toString());
        }
        System.out.println("\nDone");
    }
}
```

Read Functions

This section walks through a sample C# application. Web service client access helper classes and stubs were generated using Microsoft Visual Studio 2005. For more information about generating these classes, see Getting Started. This example demonstrates the following functions:

1. Login to the OpenAir Web Services using user-supplied credentials.
2. Perform a read of all Envelopes in the user's account using the "all" method of the Read function.
3. Perform a read of a single envelope with a user-supplied ID. This portion uses the "equal to" method of the Read function.
4. Log out of the OpenAir WebService.

Read Code Example

```
using System;
using SoapApplication.OpenAir;

namespace SoapApplication
{
    class Program
    {
        /// <summary>
        /// Instance of our OpenAir web service proxy object
        /// </summary>
        private static OAirServiceHandlerService m_svc;

        /// <summary>
        /// Prompts the user for input from the console
        /// </summary>
        private static string GetUserInput(string prompt)
        {
            Console.Write(prompt);
            try
            {
                return Console.ReadLine();
            }
            catch (System.IO.IOException e)
            {
                return null;
            }
        }

        /// <summary>
        /// Logs into the OpenAir web service using the supplied login credentials.
        /// </summary>
        /// <returns>True if successful, false if not</returns>
        private static bool Login()
        {
            // setup our login information
            OpenAir.LoginParams lp = new OpenAir.LoginParams();
            lp.api_key = "******************";
            lp.api_namespace = "company_namespace";
            lp.user = GetUserInput("Enter username: ");
            lp.password = GetUserInput("Enter password: ");
            lp.company = GetUserInput("Enter company: ");

            // get an instance of our service and login
            m_svc = new OAirServiceHandlerService();
            LoginResult loginResult;
```

```
            try
            {
                loginResult = m_svc.login(lp);
                Console.WriteLine("Logged in, session ID = " + loginResult.sessionId);

                // set our session header to contain the session ID so we can perform further
operations
                SessionHeader header = new SessionHeader();
                header.sessionId = loginResult.sessionId;
                m_svc.SessionHeaderValue = header;
            }
            catch (System.Web.Services.Protocols.SoapException e)
            {
                // catch any login problems and return
                Console.WriteLine(e.Message);
                return false;
            }
            return true;
        }

        /// <summary>
        /// Outputs the supplied envelope to the console.
        /// </summary>
        private static void PrintEnvelope(oaEnvelope env)
        {
            Console.WriteLine("----------------------------");
            Console.WriteLine("ID:\t" + env.id);
            Console.WriteLine("Name:\t" + env.total);
            Console.WriteLine("Date:\t" + env.date);
            Console.WriteLine("Total:\t" + env.total);
            Console.WriteLine("# receipts:\t" + env.tottickets);
            // output any other needed envelope information here
      }

        /// <summary>
        /// Reads all envelops from the OpenAir web service
        /// </summary>
        static void ReadAllEnvelopes()
        {
            // perform the read
            Console.WriteLine("\nPerforming read of ALL envelopes\n");
            ReadRequest req = new ReadRequest();
            req.type = "Envelope";
            req.method = "all";

            try
            {
                ReadResult[] results = m_svc.read(new ReadRequest[] { req });
```

```
                // iterate through our results and output them to console
                foreach (ReadResult result in results)
                {
                    // output any errors
                    if (result.errors != null)
                    {
                        foreach (oaError err in result.errors)
                        {
                            Console.WriteLine("Error "+err.code + ": " + err.comment + "\t" +
err.text);
                        }
                    }

                    // output the envelope read results
                    if (result.objects != null)
                    {
                        Console.WriteLine("Received "+result.objects.Length+" envelope(s) from
OpenAir");
                        foreach (oaEnvelope env in result.objects)
                        {
                            PrintEnvelope(env);
                        }
                    }
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("Error while reading envelopes:\n" + e);
            }
        }

        /// <summary>
        /// Read a single envelope from OpenAir
        /// </summary>
        private static void ReadSingleEnvelope()
        {
            Console.WriteLine("\n\nPerforming read using \"equal to\" method");
            oaEnvelope envelope = new oaEnvelope();
            envelope.id = GetUserInput("Enter an envelope id: ");
            ReadRequest req = new ReadRequest();
            req.objects = new oaBase[] { envelope };
            req.type = "Envelope";
            req.method = "equal to";
            try
            {
                ReadResult[] results = m_svc.read(new ReadRequest[] { req });
```

```
                // iterate through our results and output them to console
                foreach (ReadResult result in results)
                {
                    // output any errors
                    if (result.errors != null)
                    {
                        foreach (oaError err in result.errors)
                        {
                            Console.WriteLine("Error " + err.code + ": " + err.comment + "\t" +
err.text);
                        }
                    }

                    // output the envelope read results
                    if (result.objects != null)
                    {
                        Console.WriteLine("Received " + result.objects.Length + " envelope(s) from
OpenAir");
                        foreach (oaEnvelope env in result.objects)
                        {
                            PrintEnvelope(env);
                        }
                    }
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("Error while reading envelopes:\n" + e);
            }
        }

        /// <summary>
        /// Application entry point
        /// </summary>
        static void Main(string[] args)
        {
            if (Login())
            {
                ReadAllEnvelopes();
                ReadSingleEnvelope();

                // end our session
                m_svc.logout();
            }
            Console.WriteLine("\nPress enter to exit");
            Console.ReadLine();
        }
    }
}
```

Appendix A Error Code Listing

The API returns error codes that you can use to help you identify problems specific to a query or action on a particular object. The errors are broken out by their type and you can search for one using the error code number.

Server Errors

Error Code	Short Message	More Information
0	Success	The operation was successful
1	Unknown Error	
2	not logged in	Command required a valid Auth, but Auth failed, or was left out of the request
3	too many arguments	More arguments (XML records) were passed to a command than the command accepts
4	too few arguments	Fewer arguments were passed to a command than were expected
5	Unknown Command	There is no command by that name, request failed
6	Access from an invalid URL	Please use the URL you were provided with to access the API
7	Invalid OffLine version	Please upgrade your version of OpenAir OffLine
8	Failure + Dynamic Message	The operation has failed, Please consult the Error record that was passed, this code is reserved for dynamically generated error codes
9	Logged out	Your session is no longer valid, please issue a login command
10	Invalid parameters	Invalid parameters were used, please consult documentation

CreateUser Errors

Error Code	Short Message	More Information
201	invalid company	Create the company first, then create users
202	duplicate user nick	A user with this nickname already exists, try another one
203	too few arguments	You need to specify both a Company object and a User object
204	Namespace error	Users must be created in the same namespace as the company
205	Workschedule error	Invalid account workschedule specified

CreateAccount Errors

Error Code	Short Message	More Information
301	duplicate company nick	This company nick is already in use, try another one
302	too few arguments	You need to specify both a Company and User object
303	please pick a different password	The password entered was not hard enough to guess, please pick another to continue
304	Not enabled	CreateAccount operation is not permitted

Auth Errors

Error Code	Short Message	More Information
401	Auth failed : No such company/user/pass	The combination of usernick/companynick/password doesn't exist
402	Old TB login	Internal TB error
403	No Company name supplied	N/A
404	No User name supplied	N/A
405	No User Password supplied	N/A
406	Invalid Company name	N/A
408	Bad Password	N/A
409	Account Canceled	This account has been canceled
410	Inactive user	This user has been made inactive by their administrator
411	Account conflict, contact customer service	There is a problem with the account. Contacting customer service will allow you to use the account again
412	Wrong namespace for account	This account is not associated with the namespace provided
413	Account not privileged to access API	This user is not allowed to access the API functionality
414	Temporarily unavailable	The service is temporarily unavailable, please try back in a few minutes
415	Account archived	This account is archived
416	User locked	This user has been locked, please contact your Account Administrator
417	Restricted IP address	Access is not allow from this IP address
418	Invalid uid session	The uid passed in is not valid, please login
422	LDAP server unavailable	Unable to connect LDAP server

API Login Errors

Error Code	Short Message	More Information
501	API authentication required	API access must first be authenticated
502	API authentication failed	The request element must contain key & name attributes

Error Code	Short Message	More Information
503	Invalid or missing key attribute	N/A
504	Invalid or missing namespace attribute	N/A
505	The namespace and key do not match	N/A
506	Authentication key disabled	This key has been disabled. Please contact support for more information
555	You have exceeded the limit set for the account for input objects	Please make sure to observe the limit for input data set for your account
556	XML API rate limit exceeded	The limit of requests allowed for your company has been reached

Read Errors

Error Code	Short Message	More Information
601	Invalid id/code	There isn't a record matching the id or code you asked for
602	Invalid field	N/A
603	Invalid type or method	N/A
604	Attachment size exceeds space available	Please contact your OpenAir administrator to request more space
605	Limit clause must be specified and be less than the account limit for output data	N/A

Delete Errors

Error Code	Short Message	More Information
701	Cannot delete, failed dependency check	You must first delete all the records that have an index pointing to this record
702	Invalid note	The note could not be deleted

Add/Modify Errors

Error Code	Short Message	More Information
801	Not a valid Customer ID	The Customer ID you tried to associate with this Project does not exist, or is deleted
802	This Envelope number is already taken	Please select a different Envelope number, or specify none for auto-numbering
803	This user does not have permission to modify the record	The non-administrative user is trying to modify an administrator only record
804	Not a valid Item type	The only valid types are R and M

Error Code	Short Message	More Information
805	Reference number in use	The reference number is already in use, please select a different one
806	Already accepted by signer	You cannot modify tasks or tickets that have already been accepted by a signer
807	Invalid payment type	The payment type passed is not valid (possibly inactive, or deleted)
808	Invalid note	The note you are trying to modify is not valid
809	Invalid Timesheet	The timesheet you specified for this task does not exist, or has been deleted
810	Invalid index	The index you specified doesn't exist in that table
811	Invalid predecessor	One or more IDs in the predecessor list could not be found
812	Invalid parentid	The parentid field has an id that is not valid
813	Invalid projectid	The projectid specified doesn't exist, or was deleted
814	duplicate id_number	This id_number is already in use for this project
815	Projecttask does not exist	The Projecttask you specified does not exist
816	User role/type does not exist	The role_id or type you specified is invalid
817	Invalid envelope	The envelope id specified does not exist
819	Slip cannot be deleted	This slip is part of an Invoice, and cannot be deleted
820	Envelope not open	The envelope cannot be modified because it is no longer open
821	Timesheet not open	This error is returned under the following conditions: - The timesheet cannot be modified, it has been submitted for approval. - The timesheet is has status (A/X - Approved/Archived) and internal switch allowing editing of approved timesheets is not enabled. - Timesheet is not in Open Period and user's role doesn't allow editing of timesheets outside of Open Periods. - Timesheet has status S (Submitted) and is modified by the submitter, while internal switch doesn't allow editing of submitted timesheets by owner. - Timesheet has been exported and internal switch disallowing modification of exported timesheets is turned on. - When a user who does not own a full Account Administrator role attempts to modify timesheet of another user via API.
822	Slip cannot be modified	This slip cannot be edited
823	Slip, bad invoice id	The slip is already in an invoice, and cannot be moved to another invoice
824	Must specify name or company	The Customer/Prospect must have a valid name or company
825	Invalid invoice	The invoice ID specified does not exist
826	Date is required	N/A

Error Code	Short Message	More Information
827	Reimbursements can only be applied after the envelope is approved	N/A
828	This Invoice number is already taken	Please select a different Invoice number, or specify none for auto-numbering
829	Not a valid user	The user you specified is invalid
830	Not a valid booking type	The booking type you specified is invalid
831	No startdate or enddate specified	You must specify startdate and enddate
832	Illegal date range	Startdate must be before enddate
833	Percentage not specified	Percentage must be specified
834	Hours not specified	Hours must be specified
835	Only owner can edit this project	N/A
836	Not allowed to add entity	You must have permission to add entity
837	Not a valid account currency	You can only specify a currency currently enabled for the account
838	Not allowed to have more than one current costs per user	You can only have one cost current record per user
839	base64_data must be set to add an attachment	N/A
840	Not a valid primary filter set	The primary filter set you specified is invalid
841	Invalid email	Email is a required field
842	Invalid period	Period is a required field
843	Invalid timing	Timing is a required field
844	Invalid leave accrual rule	leave_accrual_ruleid is a required field
845	Invalid task	Task is a required field
846	Cannot create non-po purchase items	Your account or role is not configured for non-po purchase items
847	Purhaseorderid must be blank	Non-po purchase items should not be associated with a PO
848	Only non_po purchase items can be added/ modified	Non-po must be set to 1
849	Another record with the same date range already exists	Overlapping records are not allowed
850	Another record already exists as a default for this user and group	Only one default record can be added for the user and group
851	Not a valid tag_group_attribute	The tag group attribute you specified is invalid

Error Code	Short Message	More Information
852	Duplicate external_id	Another record with the same external_id is already present
853	Invalid Loaded Cost parameters	When current is set to '1', start and end dates must not be filled and vise versa
854	Too many records requested	Please modify your filter parameters to limit the data returned. If using Integration Manager, please contact OpenAir support.
855	Number of commands passed in is greater than the account limit for the API	Please separate your commands into separate requests
856	Date overlaps with existing record	The start or end dates you specified overlap with those of an existing record
857	Date range exceeded maximum	The date range specified exceeded maximum allowed
858	ForexInput error	Please note the update error
859	Invalid customer id	The customer id specified doesn't exist or was deleted
860	default_for_entity and start and end dates are mutually exclusive	Cannot set default_for_entity and start and end dates for the same record
861	Invalid customer id	The customer id specified does not match the parent invoice customer id
862	Invalid project id	The project id specified is not associated with the parent invoice customer
863	Only one project per invoice	The invoice specified is already associated with a different project
864	Error while saving user workschedule	There was an error saving the user workschedule
865	Invalid workdays	Workdays must be a CSV list containing digits between 0 (Monday) and 6(Sunday)
866	Invalid workdays or workshours	Workday and workhour values are required when setting a user workschedule
867	Distinct workhours not enabled	Only one workhour value (the default) can be specified when updating a user workschedule
868	Invalid type specified	Type must be filled and one of (project, user, customer)
869	Invalid value for primary_user_filter	primary_user_filterset can only be specified for one hierarchy of project type
870	Invalid value for primary_dropdown_filter	primary_dropdown_filter can only be specified for one hierarchy of project type
871	Invalid number of read arguments supplied	The number of argument objects must equal the number of filter clauses
872	Invalid cost type	There is no cost type with specified id
873	Invalid period	Period must be specified
874	Schedule request error	Please note the update error
875	Repeat error	Please note the update error
876	Attachment too small	Attachment size is too small

Error Code	Short Message	More Information
877	Invalid project group	project_group_id specified does not exist
878	Purchaseorder not open	The purchase order cannot be modified because it is no longer open
879	Invalid purchase order	The purchaseorder_id specified does not exist
880	Invalid purchase item	Non-PO purchase items must have a positive quality
881	Invalid attachment	Specified parent ID does not exist or parent is in a different workspace
882	Invalid reference slip ID	Specified reference slip doesn't exist or was deleted
883	Invalid portfolio project ID	Specified portfolio project ID is invalid, doesn't exist or doesn't match customer
884	Invalid portfolio link	Portfolio project cannot be subordinated to another portfolio project
885	Invalid purchase item	Mandatory date is missing in purchase item
886	Project task type mismatch	Project task type invalid, or project task not defined

MakeURL Errors

Error Code	Short Message	More Information
901	The combination of uid, app, arg, and page is not valid	The values passed don't combine to represent a valid page, check the values and try again
902	A valid record could not be created from the arg passed	Check to make sure the required fields are being passed in the arg record
903	The user does not have access to that page	That combination of app, arg, and page is not valid for this user
904	This Purchaseorder number is already taken	Please select a different Purchaseorder number, or specify none for auto-numbering
905	Invalid purchaseorder	The purchaseorder id specified does not exist
906	Invalid Cost Center	The cost_centerid specified does not exist or is inactive
907	Invalid Contact	First name, Last name and email are required fields
908	Invalid Name	Please specify a valid name for the record
909	Invalid Contact	The contact must exist, and belong to the same Customer
910	Lookup record not located	One or more lookup fields specified for the record do not exist
911	No Timesheet specified	Timesheet ID must be specified to edit a task
912	Invalid type Specified	Type must be set
913	Invalid project task specified for a project	Project task must belong to a project specified
914	Invalid resourceprofile_type_id specified	An existing resourceprofile_type id must be specified

Error Code	Short Message	More Information
915	Invalid type specified	The type and resourceprofile_type_id must be provided and must match the type-id pair in an existing record in the resourceprofile_type table
916	Table specified does not have external_id field	Make sure you selected correct association
917	This Issue number is already taken	Please select a different Issue number, or specify none for auto-numbering
918	No description specified	Issue description must be set
919	Only one default issue stage is permitted	Only one issue stage may be marked as default_for_new
920	No rate card ID specified	Rate card ID must be specified
921	Job code in use for rate card	The supplied job code is already in use for the associated rate card
922	Invalid job code specified	An existing job code must be specified
923	Invalid rate card specified	An existing rate card must be specified
924	No job code ID specified	Job code ID must be specified
925	Invalid template project ID specified	A valid project ID must be supplied for the template project ID
926	Invalid value for user cost	User cost must contain a valid value
927	Invalid user cost start date	User cost start date must not be before any previous cost start date
928	Invalid project group ID for workspace user	Project group ID must contain a valid value
929	Workspace user cannot contain both project group ID and user ID	Only project group ID or user ID can be set
930	Generic flag cannot be modified	Cannot change generic resources into users and vice versa
931	Duplicate project assignment	A user can only be assigned to a project o
932	Only admin users may update proxies	Only users in the administrator role may update proxy information
933	Not a valid proxy user	The proxy user id you specified is invalid
934	Error while creating project from template	There was an error while creating a project from a template project
935	Invalid user tag start date	User tag start date must not be before any previous tag start date
936	Error while creating project group assignments	There was an error while creating project group assignments
937	Invalid agreement ID specified	An existing agreement must be specified
938	Duplicate agreement_to_project	A unique project_id and agreement_id pair must be specified
939	View is not allowed for this user	Please check that the user logged in has the required role

Error Code	Short Message	More Information
940	Dashboard view is not allowed for this project	Please check that the project is configured for dashboard view
941	Invalid timezone specified for user	Timezone string must contain a +/- sign, four digit offset, and optionally a single letter, e.g.,: -0500,+0330, +1300a

Approve/Submit Errors

Error Code	Short Message	More Information
1001	Invalid state	Record could not be submitted, because it is currently not Open or Rejected
1002	Submit/Approve error	There are errors associated with this request
1003	Submit/Approve warning	There are warnings associated with this request

Hierarchy Errors

Error Code	Short Message	More Information
1050	Invalid hierarchy node specified	Please specify a valid hierarchy node
1051	You cannot assign multiple nodes within one hierarchy	Please specify a different hierarchy node

Custom Field Errors

Error Code	Short Message	More Information
1100	Invalid value specified for a checkbox custom field	Please specify either empty string or 1
1101	Value specified is not on the list of values for this custom field	Please specify one of the valid values for this custom field
1102	Custom field could not be saved	Please check specific error descriptions
1103	Modification of the field specified is not supported	Only valuelist field can be modified at this time
1104	This custom field value is not unique	You must enter a unique value
1105	Value specified is not on the list of values in the source pick list defined for this custom field	Please specify one of the valid values from the source pick list for this custom field
1106	One or more inline custom fields failed to be updated	Please review specific errors returned

Filterset Errors

Error Code	Short Message	More Information
1300	Invalid filter set specified	Please specify a valid filter set

XML Errors

Error Code	Short Message	More Information
2001	Invalid argument passed	Please make sure to pass valid arguments

Appendix B OpenAir Data Dictionary

Refer to the following link for a complete OpenAir Data Dictionary:
http://www.openair.com/database/single_user.html

Appendix C Best Practices

Before you begin using OpenAir SOAP API functionality, ensure your OpenAir account is fully configured and in production. As you know, OpenAir provides a number of ways you can customize your company's account to meet unique business requirements. While this flexibility allows you to maximize its effectiveness for your organization, it is helpful to establish the system before you try to access the tables and data fields within it.

Note: We highly recommend that you work with your OpenAir Professional Services (PS) consultant to design the API integration. The knowledge you gain about how tables and data fields are used in your business processes will save development time on the front end and help you optimize your integration on an ongoing basis.

Build the API Integration

The OpenAir SOAP API provides tools for building a powerful integration. Take some time to plan what you want to do, design your integration and document the process, develop your integration, and test it in your sandbox account, which provides you with a safe environment. Each step is explained in more detail in the following.

Step 1: Plan What You Want To Do

Think about what you are trying to achieve in your OpenAir implementation and how the SOAP API can increase your ability to do that. Ask and answer the following questions:

- What are your critical processes? How can the API integration help you streamline them?
- What are your repetitive tasks? How can the API integration help automate those tasks?
- What will the API integration be able to do that can help your employees save time?

Step 2: Design Your API Integration

Take time to develop a document that describes your API integration. Your PS consultant can expedite this effort and help reduce development time. Gain an understanding of what you are trying to achieve so that key players in your organization can provide valuable feedback before you begin the actual development.

Step 3: Develop Your Integration

Read this document in its entirety, talk with your PS consultant, and learn about the OpenAir data model and how it is used. Links to key information are provided in Introduction to OpenAir Web Services as well as in Getting Started.

- Develop the API integration with the help of your PS consultant. Incorporate labels and terms that will both reduce confusion and enhance the integration you develop.
- Test the API integration in your sandbox account. It is crucial that you use a non-production environment until you can be sure that the integration runs smoothly without error and does not corrupt vital production data.

Optimize the API Integration

The following suggestions will help you get the most out of your API integration. Discuss them with your PS consultant to ensure you understand why they improve the efficiency and effectiveness of your integration.

Make Batch Calls

When making calls in your API integration, request and update data records in batches. Typically, we recommend that records be grouped into batches of 500, but the specifics vary depending on the context and the expected volume of data to be transacted. Even when requesting data based on filtering criteria, multiple read operations can be specified within one read request. We recommend running batch operations during off-peak hours to minimize impact on integration performance.

Make Fewer Calls

Reducing the number of calls you make to the API improves the performance of your integration. Because API calls require a call/response over the public Internet, they can consume both time and resources. Minimizing the number of calls you make increases the speed at which your API integration operates. When working with custom fields in OpenAir SOAP Web Services, we recommend including the custom fields in the wsdl file and reading and updating custom fields as part of the native object update, as opposed to using the legacy "custom equal to" methods. See Modifying Records to Set Custom Field Values. Running batch operations during off-peak hours also minimizes impact on performance.

While the API technically allows concurrent connections, running the API from multiple clients simultaneously is NOT recommended. This may cause performance deterioration due to contention on Web and database servers' resources and can affect the performance of both the API integration and user interaction.

Note: Please refer to the Limits section in Introduction to OpenAir Web Services for more information regarding possible throttling controls. Batching multiple API operations into one request and caching data locally are the best methods to avoid our servers ever triggering throttling controls.

Cache Locally

Transactional records in OpenAir contain as many as a dozen foreign keys that refer to other records in the system. When retrieving a batch of transactional records, you will often be retrieving many records with the same foreign key value. For example, you could retrieve many charges with the same project_id or many timesheet entries with the same user_id.

To optimize performance, after retrieving a batch of charges, you should construct a message to retrieve all the project records associated with those charges and then hold those project records locally, either in memory or via persistent storage to use with the next batch. When you use a persistent cache, the integration could make sure it's up to date via use of our "newer-than" filters. We recommend getting list data, caching it, and then keeping it in sync by requesting records that have changed since the previous update. OpenAir Web Services also allows you to request deleted data since the last request, which is another way to ensure your local list data cache is up-to-date.

Another way of optimizing performance is paying attention to the range of possible foreign key values for an attribute. This range of values could be very small. For example, even a very large OpenAir account may have only 3 or 4 timetypes and every time entry record will then have one of those 3 or 4 values. Once the timetype records have been retrieved, they can be held locally for an indefinite period as timetype values change infrequently and the same small set can be referenced on every time entry transaction.

Use external_ids

In addition to caching locally, you can use external_id values (as saved in OpenAir) in place of internal OpenAir ids when related data is being referenced on a OpenAir record during an modify/add operation. This often avoids the need to request the OpenAir list data in the first place. The integration logic should properly manage possible errors if the external_id was not found in OpenAir system. See Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

Use Date Filters to Limit Amount of Data Processed

Make sure you are only requesting data that is new, modified, or deleted. When requesting list elements like projects, as mentioned previously, we recommend that you keep a local cache of records. See Cache Locally. Issuing a read method call that requests records that have been added/modified and/or deleted since the previous integration run allows the integration logic to process only a small data set of changed records. By default, the "newer-than" filter uses the 'updated' date on each record, which is the timestamp appropriate for such use. For code examples, see Example VI. read not-exported Envelopes with date filter C# and Example IV. read date filter Java.

Use not-exported Filters to Limit Amount of Data Processed

Make sure you are only requesting data that has not previously been exported. For transactional exports, we recommend exporting approved entries and then marking these records in the OpenAir system as having been exported. You can configure OpenAir to lock

exported data so that it cannot be modified after the export. You can also configure OpenAir reports and lists to show records as having been exported to another system. Export child elements and mark these child elements as being exported. For example:

- Use the not-exported filter when you export Invoices and their Slips. Since slip records are the list/child element of an Invoice, you can mark each individual Slip record as being exported. Subsequent integration runs issue a read request and the "not-exported" attribute/filter only returns qualifying transactions, i.e., transactions not previously processed.
- Use the not-exported filter to export Task records for timesheet information.
- Use the not-exported filter to export Ticket records for export expense information.

For code examples, see Example V. read not exported C# and Example III. read not exported Java.

Maintain the API Integration

Before you use your API integration, there are two additional tasks to perform: set up the storage of communication logs and determine a process for upgrading the OpenAir system. Each is explained as follows.

Store Communication Logs

In the event of an API integration error, your PS consultant or OpenAir Support can help you troubleshoot the error. To do so, you need to be able to provide them with both the request code and associated response. Store a log of recent API communications as well as the exact timestamps of API requests to OpenAir servers. We recommend that you create a communication log that stores a minimum of the last seven days transactions. See Troubleshooting for information on getting help from OpenAir Support.

Upgrade With Caution

Once your API integration is tested and you move it into production, you need to determine a process for upgrading or making changes to the OpenAir system. We recommend that you do not make changes to the OpenAir production system before testing them in your sandbox account against the API integration. In particular, use care when you need to modify an object or application setting related to data or functionality that is tied to your API integration. Always test changes in your sandbox account prior to implementing them in the production system.

Troubleshooting

If you are experiencing difficulties or would like additional information, please open a support ticket and submit it through your OpenAir account. Use a support ticket to request API access.

To open a support ticket:

1. Log in to your OpenAir account.
2. Go to Support. It is a menu option on the bottom of the main drop-down list.
3. Click the link to Create a support ticket.
   Contact Customer Care form displays in a new browser window with your Company Name.
4. Type the requested information and click Submit.

   Note: An asterisk * displays after required fields.

Our support staff and engineers will work with you to find a solution to your problem.

New Features

The following summarizes the additions, updates, and enhancements to the SOAP API since the last release.

Features for November 17, 2012

- Provide support for "Require use of expense type price on receipts" option for Android devices. See cost_is_fixed

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaItem	cost_is_fixed

Features for July 14, 2012

- Allow the setting of the "Notify requester when booking is modified" field on the booking form through SOAP API. See notify_owner.
- Added error code and related information to Add/Modify error code 885. Force error on bad date in Purchase item import. Mandatory date is missing in purchase item.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	notify_owner
oaProjectbillingrule	exclude_non_billable_task
oaRevenue_recognition_transaction	portfolio_projectid
oaSlip	portfolio_projectid

Features for May 12, 2012

- Expanded the definition of the limit attribute on the read method ReadRequest argument.
- Added a reference for an internal switch to oaForexInput. You can have an internal switch enabled in your account for user defined reporting currencies.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaProject	portfolio_projectid, is_portfolio_project

Features for March 17, 2012

- Added a "generic" attribute for read commands. By default, the API returns regular users. When you add the generic attribute, the read request returns generic users.
- Added references for internal switches that affect the behavior of the API for the following complex types: oaEnvelope, oaInvoice, oaPurchase_item, oaSlip, oaTicket, and oaTimesheet.
- Added error code and related information for MakeURL error code 941. Reject oaUser add/modify requests that contain invalid time zone identifiers.
- Added clarifying information to Add/Modify error code 821. While it is returned when a timesheet cannot be modified because it was already submitted, it is also returned when other conditions exist. See oaTimesheet.
- Added error code and related information for Custom Field error code 1106. One or more inline custom fields failed to be updated

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaCustomer	created, updated, billing_code

Features for January 21, 2012

- The following complex type was added: oaSchedulebyday. Custom fields associated with oaSchedulebyday may be requested using the read method.
- Custom fields associated with oaPurchaseorder may now also be requested using the read method.
- Custom fields associated with oaRequest_item may now also be requested using the read method.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaSchedulebyday	id, date, user_id, hours, base_hours, target_hours, target_base_hours, created, updated

Features for November 19, 2011

The following complex type was added: oaRevenueStage

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	locationid
oaRevenueStage	id, name, revenue_stage_type, created, updated
oaRevenue_recognition_transaction	is_from_open_stage

Features for September 17, 2011

- The following complex type was added: oaUserWorkschedule.
- SOAP API handles all existing task rounding rules.
- Error code and related information was added for Add/Modify error code 882.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaInvoice	credit_rebill_status, original_invoiceid
oaProject	rv_approver, rv_approvalprocess
oaProjectbillingrule	category_1id, category_2id, category_3id, category_4id, category_5id
oaRevenueContainer	project_billing_rule_filter, category_1id, category_2id, category_3id, category_4id, category_5id
oaRevenue_recognition_rule_amount	category_1id, category_2id, category_3id, category_4id, category_5id
oaSlip	ref_slipid
oaTaskTimecard	category_1id, category_2id, category_3id, category_4id, category_5id
oaUserWorkschedule	id, name, userid, use_this_schedule, account_workscheduleid, workdays, workhours, created, updated

Features for July 16, 2011

- The following arguments /options were added to the makeURL method: view-invoice, dashboard-project, grid-timesheet, report-timesheet.
- Custom fields associated with oaPayment may now be requested using the read method.

- Custom fields associated with oaUser may now be returned for regular as well as generic users.

Features for May 14, 2011

- The following complex type was added: oaRevenueContainer. Enabled support for read including oaCustField and update of externalid only.
- API will not allow negative quantity on non-PO purchase items.
- Fixed an issue where email field was reset on oaContact update when value was not provided.
- Added parentid field to oaAttachment.
- Error codes and related information were added for Add/Modify error codes 880 and 881.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAttachment	parentid
oaProjecttask	default_category_1, default_category_2, default_category_3, default_category_4, default_category_5
oaRevenueContainer	id, number, date, balancing_type, total_recognized, currency, date_approved, updated, date_submitted, approval_status, total_deferred, name, acct_date, total_accrued, projectid, externalid, total_posted, created, notes, total_invoiced, customerid, exported, prefix

Features for March 19, 2011

- oaTargetUtilization records can now be added for inactive users.
- When a new customer or client is created and payment terms are not explicitly specified, default payment terms are used.
- Job_codeid can have a value of 0 in modify operations on oaProjectassign and oaProjecttaskassign.
- Short PO Purchase_item import sets the date on fulfillments to date_fulfilled (if present) or to today's date.
- Error codes and related information were added for API Login error code 555 and MakeURL error codes 914 - 915

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaCustField	never_copy
oaTask	category_1id, category_2id, category_3id, category_4id, category_5id

Features for January 22, 2011

- Enabled custom equal to support for oaPaymenttype.
- Enabled modify and delete support for oaAttachment.
- Enabled delete support for oaBooking.
- Error codes and related information were added for Add/Modify error codes 878 - 879 and Custom Field error code 1105.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaRevenue_recognition_rule_amount	cost_center_id
oaTask	acct_date
oaTicket	externalid
oaTimesheet	acct_date

Features for November 20, 2010

- The following complex type was added: oaProjectgroup.
- The following complex types were changed: oaAttribute and oaFieldAttribute.
  - oaAttribute, a table that describes an attribute, was modified to include the following children: id, name, attribute_setid, updated, created, and notes.
  - oaFieldAttribute was added and has two children: name and value. It is used to lookup foreign keys. See Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.
- Enabled add, modify, and delete support for oaAgreement_to_project.
- Enabled delete support for oaEntitytag.
- Error codes and related information were added for Add/Modify error codes 876 - 877, Make URL error codes 936 - 938, and Custom Field error code 1104.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAttribute	id, name, attribute_setid, updated, created, and notes.
oaFieldAttribute	name and value
oaProjectassign	job_codeid
oaProjectbillingrule	daily_rate_multiplier and job_code_filter
oaProjectbillingtransaction	job_codeid
oaProjectgroup	id, attributes, assigned_users, created, updated, name, notes, and active
oaProjecttaskassign	job_codeid
oaRevenueContainer	asb_which_slips
oaUprate	job_codeid

Features for September 18, 2010

- The following complex types were added: oaAgreement_to_project, oaAttributeset, and oaIssueStatus
- Enabled read support for oaAttribute.
- The following filter was added: approved-revenue-recognition-transactions.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAgreement_to_project	agreementid, attribute, customerid, projectid, active, created, and updated
oaAttributeset	id, name, attribute, notes, created, and updated
oaBooking	job_code_id
oaCustomer	sold_to_contact_id
oaIssueStatus	id, name, attribute, active, created, and updated
oaRevenue_recognition_transaction	project_billing_rule_id, job_code_id, rate, decimal_hours, hour, minute, revenue_containerid, revenue_stageid, originatingid, and offsetsid
oaSlip	projecttask_type_id, job_code_id, and payroll_type_id

Features for July 17, 2010

- The following complex types were added: oaCategory_1, oaCategory_2, oaCategory_3, oaCategory_4, and oaCategory_5.
- Enabled custom equal to support for oaRevenue_recognition_transaction.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	starttime and endtime
oaCategory_1	id, name, code, externalid, active, created, updated, and notes
oaCategory_2	id, name, code, externalid, active, created, updated, and notes
oaCategory_3	id, name, code, externalid, active, created, updated, and notes
oaCategory_4	id, name, code, externalid, active, created, updated, and notes
oaCategory_5	id, name, code, externalid, active, created, updated, and notes
oaRevenue_recognition_transaction	category_1id, category_2id, category_3id, category_4id, and category_5id

Features for May 15, 2010

Error codes and related information were added for Add/Modify error codes 871 - 875.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAgreement	acct_date
oaCustomerpo	acct_date
oaProject	attachmentid

Features for March 20, 2010

- Lookup and Modify Custom Fields without using “custom equal to” Method
  - Added the ability to lookup and modify custom fields without having to use the “custom equal to” method. This is now the default behavior when reading SOAP objects.
  - Custom fields may now also be modified inline in a single modify request with other fields. To enable this behavior, supply the "update_custom" attribute in your modify request and set it to 1. (This attribute is not necessary when performing add requests.)
- Use Mulitple Argument Objects in a Single ReadRequest
  - Added the ability to mix “equal to” and “not equal to” argument objects in a single ReadRequest.
  - Multiple argument objects can be supplied in one ReadRequest to create an AND/OR filtering logic. To use this feature:

- Modify the method parameter to include multiple “equal to” and “not equal to” methods as desired, separated by commas. For example: “equal to, not equal to”.
- Next, supply an equal number of argument objects to filter on.
- Additionally, you may precede each method by an “and” or an “or” operator. For example: “equal to, or equal to”. If no operator is supplied, a logical AND relationship is assumed.

  Note: This feature supports only one level of logical operators does not support nesting of \ operators (i.e., equal to and (equal to or equal to).

- Lookup Foreign Keys using oaFieldAttribute - oaFieldAttribute, a new complex type, was created and it has two children: name and value. It gives you the option of adding attributes to all other complex types except: oaAddress, oaFieldAttribute, oaDate, and oaModule. You can use externalid fields as a foreign key, allowing you to add, create User, modify and upsert records in a single step instead of querying a record to first obtain the internal id. To use an externalid field as a foreign key, set the id field to the externalid field value instead of internal id field value. Then, create an oaFieldAttribute object for each field that needs to be overridden and pass those as a collection of oaFieldAttribute objects. See code examples: Example ll. modify using external_id as foreign key lookup field C# and Example ll. externalid as foreign key lookup field Java.

  Warning: The wsdl file definition for oaAttribute is changed to oaFieldAttribute. If you were using oaAttribute to lookup internal id values from externalid values on related records, the old binaries referencing the old name will still work. However, if your code is generated using the newer wsdl file, you need to change the objects to oaFieldAttribute to continue using this feature.

- Enable Mobile Services - The internal switch to Enable mobile services is now required for all of the following add-on services: Offline, iPhone, Blackberry, Pocket PC, and Palm.
- Programming Fixes, Checks, and Validations
  - Enabled “custom equal to” method for Fulfillment object.
  - Established imported and exported as required fields on import for oaImportExport.
  - Allow more than one node per hierarchy on oaUser project_access_node.
  - Updating oaProjectBillingRule does not require that cost_centerid to be populated.
  - Added Add and Modify methods to oaSchedulerequest.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaActualcost	id, name, userid, date, period, currency, cost, cost_typeid, is_accrual, externalid, notes, created, updated
oaAttachment	attachmentid
oaCostcategory	id, name, active, notes, created, updated, externalid
oaCosttype	id, name, active, notes, created, updated, externalid
oaEnvelope	currency_exchange_intolerance
oaFieldAttribute	name, value
oaRepeat	id, frequency, every, end, occur_number, how_end, exclude_dow, created, updated
oaRevenueContainer	cost_centerid
oaTicket	attachmentid, currency_exchange_intolerance

Note: The attributes field is added in many complex types. Excluded are oaAddress, oaFieldAttribute, oaDate, and oaModule. Refer to OpenAir Complex Types for field names and descriptions.

Features for January 23, 2010

- Enabled read support for oaReport.
- Enabled modify and add support for oaHierarchy.
- Enabled modify, add, and delete support for oaHierarchyNode.
- Fixed issue where user workschedule was not being set when user is created through SOAP.
- Fixed the logic that requests deleted records, applying the not-exported filter against a deleted import_export record.
- Changed the way we handle invalid utf8 characters: strip them out completely instead of converting them to decimal numbers. Removed more utf8 encoding errors in the server log for accounts not configured for utf8.
- Adjusted createUser logic to more closely follow UI logic when setting user.name field.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	owner_id
oaCompany	workscheduleid (read-only field)
oaHierarchy	externalid

Object	Fields Exposed
oaHierarchyNode	available_as_column, externalid, primary_dropdown_filter, primary_user_filterset
oaReport	id, userid, name, type, thin_client_context, date_created, email_report, relatedid, created, updated

Features for November 21, 2009

- Set User Workschedule - Added the ability to set the user workschedule via the User API object or during user account creation. See oaUser.
- Check for valid project and customer on slip add.
- Allow use of default filtering mechanism when reading project_task.
- Made sure duplicate import_export records are never created, specific to upsert or add calls.
- Modified createUser routine to return error codes.
- Modified user tag update feature to ensure tags receive a valid start_date.
- Allow 0 offset in limit clause.
- Allow modification of an entity tag for inactive users.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaEnvelope	attachmentid
oaProject	pm_approver_1, pm_approver_2, pm_approver_3, payroll_type_filter
oaResourceprofile	externalid
oaResourceprofile_type	externalid
oaUser	update_workschedule, is_user_schedule, workschedule_workdays, workschedule_workhours