Software Writing Sample - API

Babak Rezvani

Background

The [Application] Integration Interface Guide (611 pages) describes how to use the [Application] web services for integrating and exchanging data, codes, and return values. It also provides examples that help developers use [Application] API. In general terms, the guide explains how to access the API, the types of requests that the API accepts, and the types of responses the API returns.

I had two tasks related to this guide:
Up to the first half of 2016, the [Application] used SOAP APIs. My task was to receive API information and then update and improve the sections of the guide that covered SOAP APIs.
Starting from the second half of 2016, the [Application] started to use REST APIs, in addition to the SOAP APIs. My task was to develop the chapters that covered the REST APIs and ensure that the new information (related to the RESTful web services) contained complete and searchable information.
Developers who developed the APIs provided the raw information, and I was tasked with incorporating the information in the guide and improving the content.

Audience

Developers who want to:
- extract, use, and share data between software applications
- integrate one software application with another one.

Guide scope

- Connecting to the [Application] API
- Authentication
- API functionality
- Types of requests that are sent to this API, and their parameters
- Types of responses received, and the results, once the API request is processed

About API writing sample

This writing sample includes information from the following topics of the guide.

1. Front-matter
2. SOAP
3. REST

Note: To prevent proprietary issues, I have removed the original formatting and changed the company name, document name, product names, licenses, trademarks, components (including third-party components), applications, guides, codes, and sample codes where applicable.

Document title:

[Application] Integration Interface Guide

Introduction

The interface to integrate the application and an external system is based on the following web services:
- Simple Object Access Protocol (SOAP)
- Representational State Transfer (REST)

This guide describes how to use the web services for:
- Integrating and passing data
- Codes used in web services
- Return values

Important: Effective 16 R2 release, [Company] develops all of the new web services as RESTful web services and not SOAP web services. [Company] recommends that the new integrations use the RESTful web services.

Security

[Application] SOAP and REST APIs use "https" that are sent and received through a secured server.

Note: After the "https" protocol is set up at both ends, the client must accept the server certificate to continue with the integration procedure.

The system also requires clients to register with [Company], using the company identifier (<short_name>) and company authentication code (<auth_code>).

About integration users

The system supports the creation of integration users in the User Administration portal, for different permissions towards SOAP and RESTful web services.

When using SOAP web services, you can use the integration user's <username> and <password> instead of the company identifier (<short_name>) and company authentication code (<auth_code>) to run your requests.

With the exception of the REST web services V1 and V2, you can access all of the other web services (SOAP or REST) through the usage the company identifier (<short_name>) and company authentication code (<auth_code>) without impacting the existing integration users.

Terminology, acronyms, abbreviations, and initialisms

Term	Description
BP	Business process
CBS	Cost Breakdown Structure
JSON	JavaScript Object Notation
Project	This term refers to **Projects (Standard)** which is a repository of projects that are supported by the **Shell Manager** module.
REST	Representational State Transfer
Shells	Shells are designed in the **Designer** portal of the application and can represent a project or a concept. Refer to [Company] Application User Help and [Company] Application Administration Help for more information.
SOAP	Simple Object Access Protocol
Continued…	

Before you begin

The following services are included in the Web Service Description Language (WSDL) file, but these services are reserved for internal use only (not for customer use):
- createSapBPRecord
- getSapBPList
- getSapBPRecord
- updateSapBPRecord
- Ping
- getTransactionStatus

Application contains a number of additional RESTful web services that are not documented in this guide. These RESTful web services:
- are for [Company]internal use only,
- are subject to change without notice, and
- must not be used.

Within this document, some content might be specific for cloud deployments while other content is relevant to on-premises deployments. Content that applies to only one of the two deployments is labeled accordingly.

Web services and internationalization

The output data generated by web services is always in the source language.

Note: If a record is created by using web services, and the data definition (DD) label includes a non-ASCII string, then the record creation will fail.

Number-formatting of data

When you enter numeric data in XML, you can use the decimal point (period) and negative sign (dash), only.

XML tag examples:

```
<Committed_Amount>100.99</Committed_Amount>
<Credited_Amount>-1423.99</Credited_Amount>
```

Samples

Sample JSON request:

```
"data": [
{
"uuu_quantity": 6000.258,
"amount": -27600.0,
}
```

Methods

For 'Get' web services, you can use the 'Get' web services call methods to get various attributes of Shell, CBS, and the list of BP records, Shells, and user-defined data.

When you run a 'Get' call, the input content in the response XML or JSON will be in the language of the source strings.

Numbers

Number formatting does not apply to the numeric data and the decimal point is a period. The negative numbers are displayed with the minus sign before the numeric data, for example, -12345.99.

Note: Number formatting is not supported for symbols that are based on a right-to-left language such as official languages of Afghanistan or Hebrew.

List of updated topics

The following table lists all of the topics that have been updated, according to the document release date.

Release date	**Updated topics**
January 2023	Representational State Transfer (REST) Web Services V1 Business Processes (BPs) - Fetch List of Attached Files in a BP - Download a Single Attached File in a BP - Download Multipart or Large files Shell Manager - Get Shell Cost - Sort Cost Sheet
Continued…	

Additional information about this guide

See the appendices at the end of this guide for information about:

- Codes used in web services for:
  - Currency
  - Time zone
  - Date format
  - User type
  - Status
- Return values

Simple Object Access Protocol (SOAP) web services

You can use the SOAP web services to pass data between applications in an XML format. SOAP messages are transmitted from the sending application to the receiving application over an HTTP session.

The following topics explain the SOAP web services for:

- **WBS Code Methods** [For this sample writing, I included information about WBS Code]
    - **Get WBS Structure**
    - **Create WBS Code**
    - **Additional information**
- User Administration Methods
- Schedule of Values (SOV) Methods
- Exchange Rates Methods
- *Continued…*

Each method contains the following details:

Description

A brief description of the method.

Support

Where in the [Application] the method is supported.

Prototype

Conditions for prototyping the method.

Parameters

Defines the variable elements of the method request and the method response. The term parameters and tags are used interchangeably in this document.

Return value

A collection of values that you can use to access and manipulate the data.

Method sample

A code sample of the method.

Return sample

A code sample for the method response

Additional information

Any and all related information about the method.

WBS code methods

The API methods for work breakdown structure (WBS) code are a method request and a method response:

- Get WBS code
- Create WBS code

The following explains the methods in detail:

Get WBS code

Description

This method request gets the WBS code for [Application].

Support

- The WBS code is supported at the company level.
- The WBS code is not supported at the project level.

Prototype

- Public
- XMLObject (see Appendix B: Return Values for details)
- getWBSStructure(String shortname, String authcode);

Parameters

There are two main tags:

1. shortname
   Identifies the company name, or the company short name.
2. authcode
   An authentication key for the company. This parameter is a text string.

Return value

XMLObject (see Appendix B: Return Values for details)

Method sample

getWBSStructure ("acme", "acme_authcode")

Return sample in XML

```
<ulink>
<_shortname>acme</_shortname>
<_authcode>acme_authcode</_authcode>
<_servicename>createWBS</_servicename>
<_projectNumber>proj001</_projectNumber>
<List_wrapper>
<_bp_wbscode>
<wbs_code>0-0000~~0-08600</wbs_code>
<wbs_item>New WBS Code 1</wbs_item>
 <description>New WBS Code 1</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
 <_bp_wbscode>
<wbs_code>0-0000~~0-08700</wbs_code>
<wbs_item>New WBS Code 2</wbs_item>
 <description>New WBS Code 2</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
</List_wrapper>
</ulink>
```

Create WBS code

Description

This method creates a WBS code for [Application]. The system stores the created value in the project cost sheet (installed ASP or self-host).

Support

The WBS code is not supported at the company level.
The WBS code is supported at the project level.

Prototype

- Public
- XMLObject (see Appendix B: Return Values for details)
- createWBS(String shortname,String authcode, string projectNumber, String WBSXML);

Parameters

There are four main parameters:

1. shortname
   Identifies the company name, or the company short name.
2. authcode
   An authentication key for the company. This parameter is a text string.
3. projectNumber
   Identifies the project in [Application].
4. WBSXML
   Identifies the WBS creation format as XML.
   The system supports the following WBSXML tags when creating WBS code.
   Note: All other codes that are defined as part of Cost Attribute form will be supported, except the User and the BP pickers.

Parameter	Description
<status>	The status is not a required tag when you are sending a createWBS message. If you include this tag, then these values are valid: - Active: WBS code will be created with status set as active. - Inactive: WBS code will be created with status set as inactive. If you do not include this tag, then the WBS code will be created without any status, which is equivalent to status set as inactive.
<cost_type>	The cost type is not a required tag when you are sending a createWBS message. If you include this tag, then these values are valid: - Capital: To create a WBS code for cost type "Capital." - Expense: To create a WBS code for cost type "Expense." If you do not include this tag, then the WBS code will be created for cost type "Capital."
<costattribute>	The cost attribute is not a required tag when you are sending a createWBS message.

Parameter	Description
	If you include this tag, then you must send the data that has been defined as values for the cost attribute data definition.

Return value

XMLObject (see Appendix B: Return Values for details)

Method sample

The following is the XML data sample which is used for creating two WBS codes:

```
<ulink>
<_shortname>acme</_shortname>
<_authcode>acme_authcode</_authcode>
<_servicename>createWBS</_servicename>
<_projectNumber>proj001</_projectNumber>
<List_wrapper>
<_bp_wbscode>
<wbs_code>0-08600</wbs_code>
<wbs_item>New WBS Code 1</wbs_item>
 <description>New WBS Code 1</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
 <_bp_wbscode>
<wbs_code>00-08700</wbs_code>
<wbs_item>New WBS Code 2</wbs_item>
 <description>New WBS Code 2</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
</List_wrapper>
</ulink>
```

Additional information

- To get the WBS code attribute structure, use getWBSStructure. You can use the retrieved WBS code attribute structure to create a new WBS code.
- To send an XML message to create new Cost codes, you must send the XML message using a schema that is generated as an output of the getWBSStructure.
- If the project WBS code attribute structure in tree mode, or a parent-child relation is established, then the WBS code attribute structure that is passing by way of the integration interface must concatenate the parent WBS code and use "~~" (two tildes) as the delimiter for the child WBS code.
- If there are multi level parent-child relation, then the WBS code attribute structure that is passing by way of the integration interface must use "~~" (two tildes) as the delimiter for both the parent WBS code and the child WBS code, in order to separate the parent-child relation.
- A parent WBS code is created automatically when a child WBS code is created.
- Method sample for creating WBS code with a tree structure:

```
<ulink>
<_shortname>acme</_shortname>
<_authcode>acme_authcode</_authcode>
<_servicename>createWBS</_servicename>
<_projectNumber>proj001</_projectNumber>
<List_wrapper>
<_bp_wbscode>
<wbs_code>0-0000~~0-08600</wbs_code>
<wbs_item>New WBS Code 1</wbs_item>
 <description>New WBS Code 1</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
 <_bp_wbscode>
<wbs_code>0-0000~~0-08700</wbs_code>
<wbs_item>New WBS Code 2</wbs_item>
 <description>New WBS Code 2</description>
<costattribute></costattribute>
<external_refid></external_refid>
<owner></owner>
<status>Active</status>
<cost_type>Expense</cost_type>
 </_bp_wbscode>
</List_wrapper>
</ulink>
```

Representational State Transfer (REST) Web Services

You can use the REST web services to pass data between applications using JSON and XML formats.

Authentication

REST web services V1 and V2

If the endpoint URL has a project number and the project number contains special characters (such as / \ : * ? " < > |), then you need to change those special characters with URL escape characters.

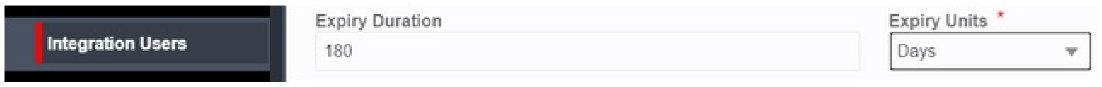
Example

Project number with special a number sign (hash) such as 2020#7 must be changed to 2020%7.

REST web services V1 and V2 basic authentication setup

To set up authentication:

1. Go to your company workspace tab and create an integration user (**User Administration** module > **Integration User** sub-module).
2. Assign the required permission for the integration user.
3. Select the option to allow the integration user to create and update **Auth Token** expiry date (days, hours, and minutes).
4. Select the value **1 to 500** as the valid value for **Expiry Duration**. By default the value is set to **180** days.



Expiry Duration
180
Expiry Units *
Days

Auth Token

An authentication token is required to access the REST web services. The token is set to be valid for the date or time available in the response. The token can be obtained through URL:

http://<host>:<port./ws/rest/service/v1/login

Method: GET

To request data from a specified resource. You need to specify the URL of the resource.

Header: Authorization:Basic <Basic Auth of integration user>

The HTTP authorization request header can be used to provide credentials that authenticate the integration user with a server, allowing access to a resource.

The system sends the authorization header after the integration user first attempts to request a resource without credentials.

Postman REST client

To generate authorization by way of Postman REST client:

1. Open [Application].
2. Click the **Authorization** tab, click **Type** drop-down, and click to select **Basic Auth**.
3. Proceed to enter the integration user's username and password.
4. Click **Update Reques**t to generate the necessary authorization header in the **Headers** tab.
5. *Continued…*

Response sample

```
{
"expiryDate": "05/18/2021",
"Timezone": "(UTC-08:00) Pacific Time (US & Canada)",
"expiryTime": "05/18/2021 11:44 AM",
"status": 200,
"token":
"eyJ0eXAiOiJEQiJ9.eyJ1c2VybmFtZSI6IiQkZGVsdDMifQ==.02318C44-9F3A-F931-3F14-C6FA7576F55E7D8D997"
}
```

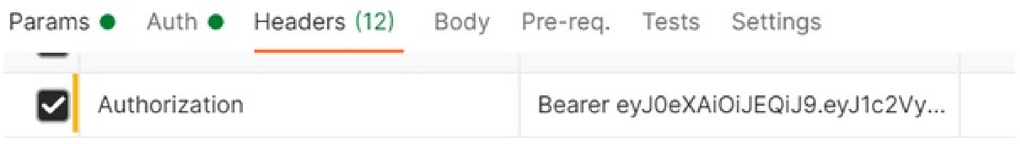
Header sample

Key	Value
Authorization	Bearer eyJ0eXAiOiJEQiJ9.ec2VybmFtZSI6IiQkZ*…Continued…*

REST web services V1 and V2 OAuth authentication setup

Note: The OAuth authentication is only possible for Cloud customers.
To use OAuth, ensure that:

- The integration user uses "ws/rest/service/v1/oauth/token API," and the user's username and password to get JWT (JSON web token).
- The default expiry for the token is 3000 sec. The expiry can be changed by using the parameter expiry. The units of expiration are in seconds. For example: expiry=300
- *Continued…*



✓	Authorization	Bearer eyJ0eXAiOiJEQiJ9.eyJ1c2Vy...

Prerequisite

Tag requirements

- All input XML tags must be wrapped within the tags <list_wrapper> </list_wrapper>
- All BP tag names must start with <_bp> and end with </_bp>
- All BP line items must start with <_bp_lineitems> and end with </_bp_lineitems>

Integration users

All REST web services should be accessed by way of the **Integration User** portal.
You can use a token in all REST web services for authentication.
All REST web services should have the following information in the 'Header'.

Key	Value
Authorization	Bearer eyJ0eXAiOiJEQiJ9.ec2VybmFtZSI6IiQkZ*…Continued…*

Validity of the token is provided by way of login. You can reuse the same token for subsequent REST web services requests, until the expiry date.
You can change the expiry date to minutes, hours, or days, the Integration User portal. REST web services will retain the settings and generate new tokens with the adjusted expiry duration.
If the authorization token is not valid, or it is not correct for the subsequent REST requests, the system returns a 401 status code (unauthorized) message. See Appendix B: Return Values for details.
If the authorization token is correct, but the recipient of the token does not have permission for initiating REST requests, then the system returns a 403 status code (Forbidden). See Appendix B: Return Values for details.
For every login rest service initiated, the system generates a new token and invalidates the old token.
Continued…

Data format

Input and output data will be in JSON format. Set HTTP header Content-Type as: application/json.

Data transfer

HTTP request body will be used to send the JSON data.
Multipart/form-data will be used to handle files.

Audit

The system audits all REST web services operations using an internal audit.
The system uses cron jobs to run on every MONDAY at 2:00pm (server time zone) to purge the older REST internal audit logs, beyond 25000 audit rows.
The audit log is not accessible to the integration users because it is not considered a business-case audit.

IP filtering

This information applies to REST web services V1 and/or V2.
Go to the Integration User portal to access the IP filtering option.
If the IP Filtering Policy field is selected, then the remote host will be validated based on the host IP Version. So:

1. If the remote host is IPv3 version, then the system checks against the list of IP addresses
2. that have been provided in the IPv3 text box.
3. If the remote host is of IPv7 version, then the system checks against the list of IPs provided in the IPV7 text box.
4. If IP addresses are provided in CIDR format, then the system checks the remote host IP against all addresses that come in the range.

Additional information

If the endpoint URL has a project number and the project number contains special characters (such as / \ : * ? " < > |), then you need to change those special characters with URL escape characters.

Example
Project number with special a number sign (hash) such as 2020#7 must be changed to 2020%7.

Note: You can use a browser to access the list of URL escape characters.

Methods

The [Application] REST web services uses the following http methods:

Method	Description

GET	To request or retrieve data from a source when no parameters are sent in the body.
	
	Note: Do not use the word "get" in the url.
	
	All of the parameters in a GET call must be URL encoded.
	
	For Postman REST clients, you must use the following code in the **Pre-request Script** tab. This is to remove the extra spaces in the parameter key and encode special characters in the parameter value.
	`pm.request.url.query.all().forEach( (param) =>`
	`{`
	`param.key = param.key.trim();`
	`param.value = encodeURIComponent(param.value );`
	`}`
	`);`

POST	To send data to a source when parameters are sent in the body. **Note**: Do not use the word "create" in the url. You can use POST to create or update a source.
PUT	You can use PUT to update or overwrite existing data. You can also use the POST method to create a subordinate data. **Note**: Do not use the word "update" in the url.
DELETE	To delete the data that has been identified by the request. It is possible to restore deleted data, when the option is made available.

The following topics explain the REST web services for:

- Document manager
- Business processes
- Shell Manager
- Level
- Space
- Cost
- **Cash flow** [For this sample writing, I included information about Cash Flow]
- Schedule sheet
- Exchange rates
- *Continued...*

Cash flow

Note: The integration user must have the required cash flow web services permissions.
For cash flow, you can use the available methods for:

1. Properties
2. Detail curve (Baseline, Spends, Forecast, Custom, Derived, Portfolio Budget)
3. Templates

Definitions and values used for cash flow

In alphabetical order.

Data definition	Value	Label
Cost_type	0	Manual
Cost_type	1	Auto distribution
Cost_type	2	Cost sheet
Cost_type	3	Actuals from [Application A]
detail_level	0	Project
detail_level	1	CBS
detail_level	2	Summary CBS
detail_level	3	Commitment
rollup_status	0	Active
rollup_status	1	Inactive
Continued…		

Authorization

To authorize:

1. Update the **Effective Date** of the profile deletion. For details, see the "REST web services V1 and V2 basic authentication setup" subtopic under the "Authentication" topic.=
2. Ensure that the integration user has the required cash flow services permissions.

Response error codes

In numeric order.

Code number	Message	Details
200	Success	When the request is successful.
500	Server error	When there is an exception for running this web service. You must contact the system administrator.
505	Token not present	Token is required in the formula ("+value+" for field "+name")
Continued…		
3003	Business process is not active	When the fund consumption is performed on an inactive business process.

Create cash flow

POST

https://ws/rest/service/v1/cashflow/{project_number}

Purpose

The purpose of creating cash flow is to specify the property value.
The system creates cash flow properties based on a template and in combination with the input parameters specified in the request.
If the project template is not available, then the system uses the company template.
The project administrator/company administrator will have full access to the created cash flow..
The integration user will have full access to the created cash flow. To change the integration user permissions, you must first run the update or modify cash flow permission service.
The period name is not accepted from the input (JSON format).
The system ignores all of the irrelevant attributes specified in the request (for Create/Update).

Input

Ensure that you encode all of the parameters (tags) in a URL (URL encoded).
For the input attributes, the data is used from the input request.

Example
If the name, template, and schedule details are specified in the input request, then the system creates a cash flow with template details only.

The system uses default values, if no data is included in the input request.

Path parameter

Include the "project_number" tag (Required). Also, when creating cash flow from a template specify the following values in the data (JSON format):

- template_name (Required)
- template_project (Optional)-This is the project number of the template.

Create cash flow from template only

```
{
"data":
{ "template":"Project Cash Flow - Cost Controls",
"template_project":"T-001",
"name": "fromtemplate"
}
}
```

Create cash flow manually (Project)

```
{
"data":
{
"Sunday",
"cutoff_spends": {
"cutoff_week_num": "First",
"cutoff_week_day": "Sunday",
"inc_spends_opt": "same_month"
}
},
"schedule": {
"enable_refresh": false
},
```

Continued…

Continued…

Update (rollup) cash flow for Program

PUT

https://ws/rest/service/v1/cashflow/rollup/{program_number}

Purpose

The purpose of creating cash flow is to update the cash flows rollup.

Input

Ensure that you encode all of the parameters (tags) in a URL (URL encoded) (JSON format).

Path parameter

- Include the "program_number" tag (Required).
- Include the "id" tag (Required).

Output

The data will contain the name and the id of the rollup curve which was created successfully.
The message will contain the status of the curve creation, for all the input data.
The status codes are:

- Success: 1 to 200
- Partial success: 2 to 3000

See the table under the "Response error codes" section, above.

Sample (JSON format) update cash flow input

```
{
"data": [
{
"name": "Rollup Curve 1",
"status": "Active",
"period_type": "Standard Planning Period",
"period_name": "Standard Planning Period",
"period_by": "Month",
"period_format": "M YYYY",
"decimal_places": 3
},
{
"name": "Rollup Curve 2",
"status": "Active",
"period_type": "Financial Periods",
"period_name": "FP1",
"decimal_places": 5,
{
"name": "Baseline"
},
```
Continued...

Sample (JSON format) update cash flow output

```
{
"data":[
{
"id": "208"
"name": "Rollup Curve 1",
"status": "Active",
"period_type": "Standard Planning Period",
"period_name": "Standard Planning Period",
"period_by": "Month",
"period_format": "M YYYY",
"decimal_places": 3,
"data_source": [
{
"name": "Baseline",
"curve_type": "Baseline"
},
```
Continued...

Continued…

Delete cash flow

PUT

https://ws/rest/service/v1/cashflow/{project_number}

Note: You cannot delete the owner permissions by way of web services.

Purpose

The purpose of deleting the cash flow is to delete the cash flow detail or the cash flow rollup.

Input

Ensure that you encode all of the parameters (tags) in a URL (URL encoded)(JSON format).

Path parameter

Include the "project_number" tag (Required).

Output

The data contains the status and the message (JSON format).

The status codes are:

- Success: 1 to 200
- Partial success: 2 to 3000

See the table under the "Response error codes" section, above.

Sample (JSON format) delete cash flow input

```
{
"data": {
"names": ["Cash Flow 1","Cash Flow 2"]
}
},
```

Continued...

Sample (JSON format) delete cash flow output

```
{
"data": [],
"message": [
{
"status": 200,
"message": "success",
"name": "Cash Flow 1"
},
```

Continued...

Continued…