Oracle
Primavera
Unifier Integration Interface Guide

Version 20
February 2022

Contents

Introduction 11
Security 13
Integration Users 15
Unifier Web Services 17
Internationalization and Web Services 19
Web Services and Internationalization 19
Number formatting of data 19
Get Web Services 19
Resource Manager Methods 21
Create and/or update Role 21
Get Role List 22
Create and/or update Resource 23
Get Resource List 24
Space Manager Methods 25
Create Level 25
Update Level 26
Get Level List 27
Create Space 28
Update Space 29
Get Space List 30
Planning Manager Methods 31
Get Planning Item Record 31
User Administration Methods 33
Create User 33
Update User 36
Get User List 37
Update User Group Membership 38
Update User Shell Membership 39
Schedule of Values (SOV) Methods 41
Get SOV 41
Exchange Rates Methods 43
Update Exchange Rates 43
Get Exchange Rates 44
Configurable Manager Methods 45
Create Records in Code & Records-based Generic Manager 45

Update Records in Code & Records-based Generic Manager 46
Shell Methods 49
Create Shell 49
Update Shell 51
Get Shell List 53
Get Project/Shell List 54
Schedule Manager Methods 57
Create Schedule Sheet Activities from Primavera P5 and P6 XML 57
Create Schedule Sheet Activities V2 from Oracle® Primavera P6™ XML 58
Create Schedule Activities from file V2 60
Update Schedule Sheet Activities from Oracle® Primavera™ P5 and P6 XML 63
Update Schedule Sheet Activities V2 from Oracle® Primavera™ P6 XML 64
Update Schedule Activities From File V2 67
Get Schedule Sheet Activities from Unifier 70
Get a List of Project Schedule Sheets 71
Get a List of Schedule Sheet Data Mappings 72
Asset Manager Methods 75
Create Asset 75
User Defined Reports (UDR) Methods 77
Get UDR Data 77
Cost Sheet Methods 81
Get Column Data 81
Update Column Data 83
Project Methods 85
Create Project 85
Create Project with additional options 87
Business Process (BP) Methods 93
Create BP Record 93
Create BP Record (Additional Information) 96
Create Complete BP Record 101
Create Complete BP Record (Additional Information) 104
Add BP Line Item 106
Add BP Line Item (Additional Information) 108
Add Complete BP Line Item 109
Add Complete BP Line Item (Additional Information) 112
Update BP Record 114
Update BP Record (Additional Information) 115
Update Complete BP Record 115
Update Complete BP Record (Additional Information) 117
Update BP Record V2 118

Update BP Record V2 (Additional Information) 119
Get BP Record 126
Get BP Record (Additional Information) 127
Get BP List 128
Get BP List (Additional Information) 130
Get Complete BP Record 131
Get Complete BP Record (Additional Information) 132
CBS Code Methods 135
Get CBS Structure 135
Create CBS Code 136
Get CBS Codes 138
Update CBS Codes 139
Sort Cost Sheet 140
Business Process (BP) Record Data Processing Rules 143
Business Process (BP) Picker Support 145
General Validation Rules Across all Services 147
General Limitations Across all Services 149
Pickers 150
Known Issues Across all Services 151
Creating and Updating Schedule Sheet 153
Data Mapping 155
Associate CBS Codes with Activities 159
DM REST Service (Company Authentication) 161
DM REST API Using Company Authentication 161
Folder Services 161
Create Folders by Path 161
Create Folders by Parent Folder ID 162
Update Folders Meta Data by Path 163
Update Folder Meta Data by Folder ID 163
Get Folders or Documents Meta Data by Path 164
Get Folders or Documents Meta Data by Parent Folder ID 165
Document Services 165
Create Documents by Path 165
Create Documents by Parent Folder ID 166
Update Documents Meta Data by Path 167
Update Document Meta Data by Document ID 168
Get Document by Path 168
Get Documents by Parent Folder ID 169
Get Document by File ID 170

REST Web Services V1 171
Default Integration User 172
Standards 173
REST API Details in Document Manager 175
Folder Services 175
Create Folders by Path 175
Create Folders by Parent Folder ID 177
Update Folders Meta Data by Path 179
Update Folder Meta Data by Folder ID 181
Get Folders or Documents Meta Data by Path 182
Get Folders or Documents Meta Data by Parent Folder ID 184
Document Services 185
Create Documents by Path 185
Create Documents by Parent Folder ID 186
Update Documents Meta Data by Path 188
Update Document Meta Data by Document ID 189
Get Documents by Path 190
Get Documents by Parent Folder ID 192
Get Document as Tiff File 193
Get Document by File ID 194
Search Document or Folder Properties 194
Rename a Node by Node ID 197
Rename Unpublished Document by File ID 198
Get Node Permissions by Node Path and Project Number 199
Get Node Permissions by Node Path, Project Number, and User or Group 202
Add Node Permissions by Node Path and Project Number 204
Update Node Permissions by Node Path and Project Number 207
Remove Node Permissions by Node Path and Project Number 209
Remove Node Permissions by Node Path, Project Number, and User or Group 210
Set Permissions in Document Manager 211
Granting, Inheriting, and Pushing permissions to files or folders 212
Removing Permissions to Files or Folders 212
Get Permissions on Files or Folders 212
REST API Details in Business Processes 213
Fetch BP Record List 213
Get BP Record 221
Get BP Record With Attachments 224
Create BP Record 232
Update BP Record 240
Update BP Record with Attachment (REST API Details in Business Processes) 247
Create BP Record with Attachment 254
Fetch BP Record List with filter_criteria 262
REST API Details in Shell Manager 267
Create Shell 267

Update Shell 268
Get Shell 270
Get BPs in Shell or Company 272
Get Project Shell List 273
Create Project 274
REST API Details in Level 281
Authorization 281
Response Error Codes 281
Get Level List 282
Create Level 285
Update Level 287
REST API Details in Space 289
Authorization 289
Response Error Codes 289
Get Space List 290
Create Space 293
Update Space 295
REST API Details in Cost 299
Get CBS Codes 299
Get CBS Codes Sample Response 300
Create CBS Codes 301
Update CBS Codes 304
Get Column Data 306
Update Column Data 307
REST API Details in Cashflow 313
Definitions or Values Used 314
Response Error Codes (REST API Details in Cashflow) 318
Get Properties List 325
Project or Shell Sample 326
CBS Sample 330
Summary CBS Sample 334
Commitment Type Sample 338
Get Templates List 340
Project or Shell Sample 342
Create Cashflow 347
Create Cashflow From Template Only 348
Create Cashflow Manual - Project or Shell 348
Create Cashflow Manual - CBS 351
Create Cashflow Manual - Summary CBS 354
Create Cashflow Manual - Commitment 356
Create Summary Curve 358
Create Rollup Cashflows for Program or Company 360
Update Rollup Cashflows for Program or Company 363

Delete Cashflow 366
Delete Cashflow - Summary Curves 367
Get Cashflow Data 368
Get Summary Cashflow Data 372
Get Rollup Status 376
Get Rollup Status - For Template Cashflows 377
Update Rollup Status 378
Update Rollup Status - For Template Cashflows 380
Refresh Cashflow 381
Get Cashflow Refresh Job Status 383
Get Cashflow Properties 384
Get Summary Cashflow Properties 409
Get Cashflow Permissions 410
Update or Modify Cashflow Permission 412
Create (Add User or Group) Cashflow Permission 418
Delete (Remove User or Group) Cashflow Permission 422
Update Cashflow 424
Update Cashflow Data 429
Create Distribution Profiles 431
Get Distribution Profiles 434
Update Distribution Profiles 436
Delete Distribution Profiles 439
Refresh Cashflow Curves 440
Refresh CashFlow Sample 441
Get Cashflow Permissions 442
Refresh CashFlow 442
REST API Details in Schedule Sheet 445
Get Schedule Sheet Data Mapping 445
Get Schedule Sheets 446
Update Schedule Activities 447
Response Codes 448
REST API Details in Exchange Rates 449
Get Exchange Rates 449
Update Exchange Rates 450
REST API Details in Data Structure Setup 453
Authorization 453
Get Data Elements 453
Create Data Element 455
Update Data Element 457
Delete Data Elements 460
Create DDS Definition 461
Update DDS Definition 464
Delete DDS Definition 467

Get DDS Definition 468
Create DDS Data 470
Update DDS Data 473
Delete DDS Data 475
Get DDS Data 476
Get Data Definition 478
Create Data Definition 483
Update Data Definition 493
Delete Data Definition 500
Response Error Codes 501
Partner Company Services V1 503
Get Partner Company 506
Create Partner Company 508
Update Partner Company 510
Update Partner Company Shortname 511
Add (Update) Partner Company Shell Membership 511
Remove (Update) Partner Company Shell Membership 512
User Services V1 513
Get User 513
Create User 520
Update User 523
Update User Shell Membership 524
Update User Group Membership 525
User Defined Report (UDR) Services V1 525
Passing Values to UDR 526
Get Templates List 528
Project or Shell Sample 529
Data 531
Data Format 531
Data Transfer 531
REST API Details in Event Driven Notification 533
Get Event Driven Notification 533
REST Web Services V2 537
Standards 538
REST API Details in Business Processes 540
Create BP Record 540
Update BP Record 546
Update BP Record with Attachment (REST API Details in Business Processes) 553
Create BP Record with Attachment 560
Response Error Codes (REST API Details in Business Processes) 570
Funding REST API Details 571
Get Funds List 571

Create Fund 572
Update Fund 575
Delete Fund 576
Manual Fund Consumption 577
Get Fund Columns 583
Get Fund Assignment Order 585
Get Fund Assignment Order for Cost Sheet 586
Update Fund Assignment Order 588
Update Fund Assignment Order for Cost Sheet 589
Update Fund Status 590
Get Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio) 592
Update Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio) 597
Response Error Codes (Funding REST API Details) 603
Non-Workflow BP Permissions 609
Get BP Records Permission 609
Get BP Permission Sample Response 610
Update/Modify BP Records Permission 611
Update/Modify BP Records Permission Sample Response 613
Create (Add User/Group) BP Records Permission 616
Create (Add User/Group) BP Records Permission Sample Response 618
Delete (Remove User/Group) BP Records Permission 619
Delete (Remove User/Group) BP Records Permission Sample Response 620
Currency 621
Currencies and codes (A-G) 621
Currencies and codes (H-N) 623
Currencies and codes (O-T) 625
Currencies and codes (U-Z) 626
Time Zone 629
Date Format 635
User Type 637
Company Address 639
Status 641
Check Box 643
Return Values 645
Copyright 650

Introduction

The interface to integrate Unifier and an external system is based on Web Service and RESTful API methodologies. This document describes the methods required for the integration and the data being passed in those methods.

The following services are included in the WSDL file, but these services are reserved for internal use (not for customer use):

- `createOIMUser`
- `updateOIMUser`
- `createSapBPRecord`
- `getSapBPList`
- `getSapBPRecord`
- `updateSapBPRecord`
- `Ping`
- `getTransactionStatus`

Within our documentation, some content might be specific for cloud deployments while other content is relevant for on-premises deployments. Any content that applies to only one of these deployments is labeled accordingly.

Security

The Unifier Web Services and RESTful services are sent and received through a secured server using "https" protocol. Once https is set up at both ends, the client must accept the server certificate to continue with the Integration procedure.

Unifier also requires clients to register on Unifier with the `shortname` (the Company identifier) and `authcode` (the Company authentication code).

> **Note**: Web Services calls can only be made using Owner Company details.

Integration Users

Unifier supports the creation of integration users in the User Administration for different permissions towards REST and SOAP services.

The username and password for these integration user can be used in SOAP services, as well as for DM REST API Using Company Authentication services instead of short name and auth code to run the requests.

> **Note**: All the other Web services, including SOAP and Rest but excluding V1 and V2, will still be accessible by Company shortname/auth code, and do not have any impact on existing end-users.

Unifier Web Services

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>

Terminology

- Project: This term refers to Projects (Standard), which are projects that are supported by Unifier without going through the Shell Manager.
- Shells: Shells are designed in uDesigner and can represent virtually anything you want. For example, a shell can represent a building, a project, or a concept such as "innovative ideas." Refer to Unifier Admin and User Guide for more information.
- Shell with Cost Code CBS: These are shell types for use with the CBS Cost Manager.
- Shell with Cost Code Generic: These are shell types with for use with the Generic Cost Manager.

Internationalization and Web Services

About Web Services

New records can be created and line items added using Integration through Web Services. Also, the Unifier Schedule Manager integrates with Primavera scheduling software by way of Web Services.

> **Note**: Integration through Web Services must be coordinated with an Oracle Primavera representative.

As Project Administrator, you can receive email notification of the successful creation of a shell instance, for shells that are created manually, through Web Services or a CSV file upload, or through auto-creation. This notification can be set up in email notifications in uDesigner. Also, you can set your **Preferences** to control whether you receive these notifications.

Refer to the *Unifier Reference Guide* for data elements you can use with Web Services.

Web Services and Internationalization

The output data generated by Web Services is always in the source language.

> **Note**: If a record (Example: Business Process) 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.

Examples

XML Tag: <Committed_Amount>100.99</Committed_Amount>

XML Tag: <Credited_Amount>-1423.99</Credited_Amount>

Get Web Services

You can use the Get Web Services call methods to get various attributes of Shell, CBS, and the list of Business Process records, Shells, and User defined data.

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

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.

Resource Manager Methods

In This Section

Create and/or update Role 21
Get Role List 22
Create and/or update Resource 23
Get Resource List..... 24

Create and/or update Role

Description

This method creates (and updates) Resource Manager Roles.

Support

This method can only support creating and updating Roles at the company level.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Prototype

public XMLObject createUpdateRole (String shortname, String authcode, String rolesXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
rolesXML	Information of the role(s) that must be created and/or updated

Return Value

See Return Values

Sample Method

createUpdateRole("acme", "acme_authcode", "XML")

Additional Information

_role tag can be repeated to create multiple roles.

Role name will be used as a unique identifier to update existing role information.

Value of <role_currency> element should be valid. For a list of valid country names, see Currency.

Get Role List

Description

This method gets a list of roles created at company level.

Support

This method is only supported at company level.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Prototype

public XMLObject getRoleList (String shortname, String authcode, String filterCondition);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
filtercondition	Condition based on which list of roles can be filtered

Return Value

See Return Values

Sample Method

getRoleList("acme", "acme_authcode",uuu_role_status=Active)

Additional Information

Only one filter condition can be provided at a time.

Only operator available for filter condition will be "=".

Example:

If user wants to get a list of roles with status Active then filterCondition should be <filterCondition>uuu_role_status = Active</filterCondition>

If user wants to get a role with name "Architect" then filterCondition should be <filterCondition>uuu_role_name = Architect</filterCondition>.

Create and/or update Resource

Description

This method creates (and updates) Resource Manager Resources.

Support

This method can only support creating and updating Resources at the company level.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Prototype

public XMLObject createUpdateResource (String shortname, String authcode, String resourceXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
resourceXML	Information of the resource(s) that will be created and/or updated

Return Value

See ***Return Values*** (on page 645).

Sample Method

createUpdateResource("acme", "acme_authcode", XML)

Additional Information

_resource tag can repeat to create multiple resources.

Get Resource List

Description

This method gets a list of resources created at company level.

Support

This method is only supported at company level.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Prototype

public XMLObject getResourceList(String shortname, String authcode, String filterCondition);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
filtercondition	Condition based on which list of roles can be filtered

Return Value

See ***Return Values*** (on page 645).

Sample Method

getRoleList("acme", "acme_authcode",uuu_resc_status=Active)

Additional Information

Only one filter condition can be provided at a time.

Only operator available for filter condition will be "=".

Example:

If user wants to get a list of resources with status Active then filterCondition should be <filterCondition>uuu_resc_status = Active</filterCondition>

Space Manager Methods

In This Section

Create Level 25
Update Level 26
Get Level List 27
Create Space 28
Update Space 29
Get Space List 30

Create Level

Description

This method creates level record in Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject createLevel (String shortname, String authcode, string projectNumber, String levelXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
levelXML	XML that contains information related to level that is getting created

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

createLevel ("acme", "acme_authcode", "proj-01", levelXML) will return the XML Record of the Level created.

Additional Information

Multiple levels can be created by repeating <_level> for each level record.

This service can be used for both Shells of cost code type CBS and Generic.

XML tags under _level element are based on the integration interface definition of level form.

Update Level

Description

This method updates level record in Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject updateLevel (String shortname, String authcode, string projectNumber, String levelXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
levelXML	XML that contains information related to level that is getting created

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

updateLevel ("acme", "acme_authcode", "proj-01", levelXML) will return the XML Record of the Level updated.

Additional Information

Multiple levels can be updated by repeating <_level> for each level record.

This service can be used for both Shells of cost code type CBS and Generic.

XML tags under _level element are based on the integration interface definition of level form.

To upate a level record in Unifier, value under <uuu_sp_level_name> will be used as identifier.

Get Level List

Description

This method will get a list of level record from Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject getLevelList(String shortname, String authcode, string projectNumber);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

getLevelList ("acme", "acme_authcode", "proj-01") will return the XML Record of the Level records.

Additional Information

XML tags under _level will be based on integration interface definition of level form.

This service can be used for both Shells of cost code type CBS and Generic.

Create Space

Description

This method creates space record of a space type in Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject createSpace (String shortname, String authcode, string projectNumber, String spacetype, String spaceXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
spacetype	Identifier of the space type
spaceXML	XML that contains information related to space that is getting created

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

createSpace ("acme", "acme_authcode", "proj-01", "Useable Spaces", spaceXML) will return the XML Record of the Level created.

Additional Information

Multiple spaces can be created by repeating <_space> for each space record.

This service can be used for both Shells of cost code type CBS and Generic.

XML tags under _space element are based on the integration interface definition of space form.

uuu_sp_level_picker element value will be used to identify the level under which the space record should be created.

Update Space

Description

This method will update space record in Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject updateSpace(String shortname, String authcode, string projectNumber, String spacetype, String spaceXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
spacetype	Identifier of the space type
spaceXML	XML that contains information related to space that is getting updated

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

updateSpace ("acme", "acme_authcode", "proj-01","Useable Spaces", spaceXML) will return the XML Record of the Level updated.

Additional Information

Multiple spaces can be updated by repeating <_level> for each space record.

This service can be used for both Shells of cost code type CBS and Generic.

XML tags under _space element are based on the integration interface definition of space form.

To update a space record in Unifier, value under <uuu_sp_level_picker> and data element based on data definition Sys Space Name will be used as identifier.

Get Space List

Description

This method will get a list of space record from Unifier. This works only for project or shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject getSpaceList(String shortname, String authcode, string projectNumber);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.

Return Value

XMLObject (XMLObject is described at the beginning of this document)

Sample Method

getSpaceList ("acme", "acme_authcode", "proj-01") will return the XML Record of the Level records.

Additional Information

XML tags under _space will be based on integration interface definition of level form.

This service can be used for both Shells of cost code type CBS and Generic.

Planning Manager Methods

In This Section

Get Planning Item Record 31

Get Planning Item Record

Description

This method gets a specific Planning Item record in Unifier. This works for both company level and project or shell level Planning Items.

Support

This process supports all type of Planning Items.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject getPlanningItem (String shortname, String authcode, string projectNumber, String BPName, String planning Item);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier. Pass null for Company Level Planning Items
BPName	identifier of the planning item type in Unifier, for example: Capital Projects
planningItem	identifier of the planning item example: Tower

Return Value

See ***Return Values*** (on page 645).

Sample Method

getPlanningItem("acme", "acme_authcode", "proj-01", "Capital Projects","Tower") will return the XML Record of the Planning Item.

Additional Information

If the planningItem is left blank or invalid planningItem then the method will returns the Planning Item's skeletal XML.

To get company level Planning Item record information, send null for projectNumber parameter. If you send a project number for a Planning Item type that is at Company level then you will get an error that Planning Item type is not correct.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

User Administration Methods

In This Section

Create User 33
Update User 36
Get User List 37
Update User Group Membership 38
Update User Shell Membership 39

Create User

Description

This public method will allow users to create one or more company users.

> **Note**: If you access Unifier through OIM, you will not be able to create user.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject createUser(String shortname, String authcode, String copyFromUserPreferenceTemplate, String userXML);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication key for the company, in text string
copyFromUserPreferenceTemplate	Name of the template from which user preference should be copied over when new user is created

Parameter	Description
userXML	▸ If the User Attribute Form is defined, it should be based on the integration interface definition of the user attribute form in udesigner. ▸ If the User Attribute Form is not defined, the element of userXML will be from the table in the following "userXML Elements" section, below.

userXML Elements (When User Attribute Form is not Defined)

Tag Name	Required	Description	Valid values
<username>	Yes	User name that is going to be used by user to sign in to Unifier.	
<password>	Yes	User's password	
<firstname>	No	First name of the user.	
<lastname>	No	Last name of the user.	
<title>	No	Title information	
<email>	No	Email of the user	
<workphone>	No	Work phone number of the user	
<homephone>	No	Home phone number of the user	
<fax>	No	Fax number of the user	
<cellphone>	No	Cell phone number of the user	
<pager>	No	Pager number of the user	

Tag Name	Required	Description	Valid values
<status>	No	Status of the newly created user (Active, Inactive, On-Hold)	Active: 1 Inactive: 0 On-Hold: 2
<timezone>	No	Time zone of the user.	See Time Zone. You have to send the data from value column, for example, for time zone GMT -11:00, Samoa Standard Time (Samoa) send 10.
<dateformat>	No	User preference date format.	See Date Format. You have to send the data from value column, for example for MM/DD/YYYY HH:MM AM send 0 (zero).
<usertype>	No	Defines the user type, based on license terms.	0 for Standard User 1 for Portal User

Return Value

See ***Return Values*** (on page 645).

Sample Method

createUser("acme", "acme_authcode", "User Pref Temp 1", "<user information>")

Additional Information

If the user preference template is provided then the user preference will be set based on the user preference template on the newly created company user.

'_user' will be the base tag which can be repeated with the data in order to create multiple company users.

The Web Services call for user administration includes a XML tag to support the Proxy login setup. The Data Element (DE) for the XML tag is: "`uuu_user_proxy_config`."

This is not a required field, so the default value remains unchecked.

Users can be created using the CreateOject WS method, and the XML tag is: "`allowproxy`."

To map the values for checkbox:

- If `uuu_user_proxy_config` or `allowproxy` is set to Yes, then the Check box “Do not allow Proxies” in Proxy tab should be unchecked.
- If `uuu_user_proxy_config` or `allowproxy` is set to No, then the Check box “Do not allow Proxies” in Proxy tab should be checked.

Update User

Description

This public method will allow users to update one or more company users.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject updateUser(String shortname, String authcode, String userXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
userXML	▸ If the User Attribute Form is defined, it should be based on the integration interface definition of the user attribute form in udesigner. ▸ If the User Attribute Form is not defined, the element of userXML will be from the userXML Elements table in the "Create User" section above.

Return Value

See ***Return Values*** (on page 645).

Sample Method

updateUser("acme", "acme_authcode", "<user information>")

Additional Information

'_user' will be the base tag which can be repeated with the data in order to update multiple company users.

The unique identifier will be the user name and based on the matching user name, the passed on fields should be updated.

The Web Services call for user administration includes a XML tag to support the Proxy login setup. The Data Element (DE) for the XML tag is: "`uuu_user_proxy_config`."

This is not a required field, so the default value remains unchecked.

Users can be created using the CreateOject WS method, and the XML tag is: "`allowproxy`."

To map the values for checkbox:

- If `uuu_user_proxy_config` or `allowproxy` is set to Yes, then the Check box “Do not allow Proxies” in Proxy tab should be unchecked.
- If `uuu_user_proxy_config` or `allowproxy` is set to No, then the Check box “Do not allow Proxies” in Proxy tab should be checked.

Get User List

Description

This public method will allow users to get a list of company users.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject getUserList(String shortname, String authcode, String filterCondition);

Parameters

Parameter	Description
Shortname	identifier of the company, company’s short name
Authcode	authentication key for the company, in text string
filterCondition	Information or condition to filter users list

Return Value

See ***Return Values*** (on page 645).

Sample Method

If User Attribute Form is defined:

getUserList("acme", "acme_authcode", "uuu_user_status=0 or uuu_user_status=1 or uuu_user_status=2");

If User Attribute Form is not defined:

getUserList("acme", "acme_authcode", " status=0 or status=1 or status=2");

Additional Information

This method will allow users to get a list of company users based on the matching filter condition on either Username or First Name or Last Name or Email or Status. The XML element for these fields will be based on whether the User Attribute Form has been defined, or not, as shown below:

Field	When User Attribute Form is Defined	When User Attribute Form is not Defined
Username	uuu_user_loginname	username
First Name	uuu_user_firstname	firstname
Lastname	uuu_user_lastname	lastname
Email	uuu_user_email	email
Status	uuu_user_status	status

Only one filter condition can be provided at a time using the "=" operator.

Method will return the list of Users with matching conditions.

Update User Group Membership

Description

This public method will allow users to update one or more company user's membership to company level groups.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject updateUserGroupMembership(String shortname, String authcode, String membershipXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string

Parameter	Description
membershipXML	Information of the user(s) that must be added and/or removed from company level groups

Return Value

See ***Return Values*** (on page 645).

Sample Method

updateUserGroupMembership("acme", "acme_authcode", "<user information>")

Additional Information

'_user' will be the base tag which can be repeated with the data in order to update multiple company users' group membership.

The unique identifier will be the user name and based on the matching user name, the passed on fields should be updated.

Update User Shell Membership

Description

This public method adds users to shell membership, updates user status in shell membership and adds/removes users to/from groups at shell level.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject updateUserShellMembership(String shortname, String authcode, String projectNumber, String membershipXML);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication key for the company, in text string
projectNumber	Identifier of project or shell in Unifier

Parameter	Description
membershipXML	Information of the user(s) that must be added and/or removed from shell groups including activation / inactivation of user under a project or shell

Return Value

See ***Return Values*** (on page 645).

Sample Method

updateUserShellMembership("acme", "acme_authcode", "Proj-0001", "<user information>")

Additional Information

The project/shell number is mandatory

'_ shelluser' will be the base tag which can be repeated with the data in order to update multiple company/shell users and their group membership.

The unique identifier will be the user name and based on the matching user name, the passed on fields should be updated.

‘group_add’, ‘group_remove’ and ‘status’ are optional tags.

Status can be either ‘Active’, ‘Inactive’ or empty.

Empty status or no status should be interpreted as ‘Active’ status.

Pre-requisite for this web service to successfully add users to the shell membership:

Add the company user to the shell membership only if the user exists in company users log at the company level and is with active or on-hold status.

Add the partner user to the shell membership if the user exists in partner user log at the company level and is with active or on-hold status. If the partner user doesn’t exists at the company level than the web service will user to both company level partner user log and to the shell membership after checking for license restrictions

Schedule of Values (SOV) Methods

In This Section

Get SOV.....41

Get SOV

Description

This method will get schedule of values information of a commitment.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject getSOV(String shortname, String authcode, String projectNumber, String BPName, String recordNumber);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication key for the company, in text string
projectNumber	Identifier of project or shell in Unifier
BPName	Identifier of the base commits business process in Unifier, for example: Contracts. This is case sensitive, and should be the name as given in uDesigner
RecordNumber	The record number is base commit record number that the SOV is associated to.

Return Value

See ***Return Values*** (on page 645).

Sample Method

getSOV("acme", "acme_authcode", "PRJ-0001", "Contracts", "CON-0001")

Additional Information

getSOV can be used to get the SOV for General Spends and Payment Applications.

Tags under <SOV_header> parent tag will change if the SOV is group by commit codes or individual line items.

Exchange Rates Methods

In This Section

Update Exchange Rates..... 43
Get Exchange Rates 44

Update Exchange Rates

Description

This method adds and updates exchange rates defined at company level.

Support

This applies to exchange rates at company level only.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject updateExchangeRates(String shortname, String authcode, String rates;

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
Rates	Exchange rates of one or more currency codes with respect to company base currency.

Return Value

See ***Return Values*** (on page 645).

Sample Method

updateExchangeRates ("acme", "acme_authcode", "XML")

Additional Information

User can add rates using Web Services call.

updateExchangeRates will always create new exchange rate record and will mark it "Active."

New currency codes can be added as part of this call.

If an invalid currency code is provided as part of currency code then that code will be ignored and other codes will be processed.

If the call is successful, new record with rates will be added.

Currency name is not mandatory. For valid currency codes see Currencies and Codes.

Get Exchange Rates

Description

This method will get exchange rates from the latest exchange rates record.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject getExchangeRates(String shortname, String authcode);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
Rates	Exchange rates of one or more currency codes with respect to company base currency.

Return Value

See ***Return Values*** (on page 645).

Sample Method

getExchangeRates("acme", "acme_authcode", rates)

Configurable Manager Methods

In This Section

Create Records in Code & Records-based Generic Manager..... 45
Update Records in Code & Records-based Generic Manager..... 46

Create Records in Code & Records-based Generic Manager

Description

This method creates records under a generic manager class. This applies only to code-& records based generic managers.

Support

This applies to code-& records based generic managers at Company and Project or Shell levels.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject createConfigurableModuleRecord(String shortname, String authcode, String projectNumber, String CMCode, String ClassName, String copyFromRecord, String recordXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of project or shell in Unifier. Value can be left empty/ignored if specified CM is a company level manager.
CMCode	Identify CM ('CM1', 'CM2', …)
ClassName	Name of the class (Buildings, Electrical, IT Equipments etc..,)

Parameter	Description
copyFromRecord	Can be left empty if not copying from another record
recordXML	content of the record that needs to be created. This is based on Integration interface design for class in uDesigner.

Return Value

See ***Return Values*** (on page 645).

Sample Method

createConfigurableModuleRecord ("acme", "acme_authcode", "Buildings","CM5", "Electrical","", "XML")

Additional Information

Same logic while creating new records from UI will be used while creating a record from integration. This includes validation logic, setting status, etc.

All validations on detail form including required fields and user defined form validations will be honored while creating new records through web services.

Once record is created, detail form will be in Finish Editing mode. Data will be rolled up to the corresponding manager sheet.

Data Element which is based on data definition viz.., SYS Numeric Logical Datasource, SYS Date Logical Datasource, SYS Business Process Datasource and SYS Project Cost Datasource will not be available as part of Integration.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Update Records in Code & Records-based Generic Manager

Description

This method updates records under a generic manager class. This applies only to code-& records based generic managers.

Support

This applies to code-& records based generic managers at Company and Project or Shell levels.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject updateConfigurableModuleRecord(String shortname, String authcode, String projectNumber, String CMCode, String ClassName, String copyFromRecord, String recordXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of project or shell in Unifier. Value can be left empty/ignored if specified CM is a company level manager.
CMCode	Identify CM ('CM1', 'CM2', …)
ClassName	Name of the class (Buildings, IT Equipments etc..,)
copyFromRecord	Can be left empty if not copying from another record
recordXML	Content of the record that needs to be created. This is based on Integration interface design for class in uDesigner.

Return Value

See ***Return Values*** (on page 645).

Sample Method

updateConfigurableModuleRecord ("acme", "acme_authcode", "Buildings","CM5", "Electrical","", "XML")

Additional Information

Same logic while creating new records from UI will be used while creating a record from integration. This includes validation logic, setting status, etc.

All validations on detail form including required fields and user defined form validations will be honored while creating new records through web services.

Once record is created, detail form will be in Finish Editing mode. Data will be rolled up to the corresponding manager sheet.

Data Element which is based on data definition viz.., SYS Numeric Logical Datasource, SYS Date Logical Datasource, SYS Business Process Datasource and SYS Project Cost Datasource will not be available as part of Integration.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic

Shell Methods

In This Section

Create Shell 49
Update Shell 51
Get Shell List 53
Get Project/Shell List 54

Create Shell

Description

This method will allow users to create shell instances under a shell type. User can create more than one shell at a time.

Support

This method only supports creation of shells under a company and is not associated with any project.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject createShell(String shortname, String authcode, String copyFromShellTemplate, String shellXML);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
copyFromShellTemplate	This will be number of another shell instance or shell template. Based on shell template, shell type will be determined to create new shell
shellXML	This will be based on Shell integration interface definition. Without this definition defined user will not be allowed to create a shell instance

shellXML Elements

Tag Name	Required	Description	Valid values
<_shell>	Yes	Base tag for new shell data. This tag can be repeated to create more than one shell instance	NA
<_module options>	No	Base tag for options.	This tag is optional. If this tag is not provided then Primavera Unifier will create shell by copying user / groups from template or shell provided under copyFromShellTemplate.
<copyModules>	No	Option to either copy modules or not.	This tag can only go as a child element under <_moduleOptions> tag. This tag is identify either to create a new shell by copying all modules including user / group or just creates a new shell by copying user / group from template provided under copyFromShellTemplate. Only valid values are Yes / No. Yes means copy all modules including user / groups. No means copy only user / groups. If tag is not provided or value is empty then default behavior will be to No.
<defaultCurrency>	No	Option to specify the currency code for the shell	A new child element of <defaultCurrency> will be available within the <_moduleoptions> tag. Valid values for the <defaultCurrency> tag are the code for the currency rather than the names. For example, USD is valid, United States Dollars is not. If the code is not a valid code then the web service will fail. If the tag is present and the value is empty then template currency will be selected.

The Document Manager attribute that is used to define the Document Management Structure (DMS) originates from the shell template that was used to create the shell using "copyFromShellTemplate" parameter. If the shell template does not contain any references to the DM attribute form, the shell that was created by using such template will have the system-defined attribute form in order to define the DMS.

Return Value

See ***Return Values*** (on page 645).

Note: The message will contain new Shell Number and Shell Name that is created within Unifier.

Sample Method

createShell("acme", "acme_authcode", "Proj-0001","<XML Content>");

Note: If you do not provide values for elements that are not required then the values from template will be used to create new shell.

Additional Information

User will be allowed to create shells of type CBS and Generic. For more information out types of shells refer to uDesigner Guide.

User will be able to pass in a value for Location Picker element while creating shell instance.

All other validation while creating shell instance manually will be honored through Integration viz.., required fields, duplicate shell id etc..,

Users will be allowed to create multiple shells using this service call. If creation of one shell fails then entire transaction will be rolled back.

All shells created will be based on same template

Shell currency will be based on the template.

Update Shell

Description

This method will allow users to update shell instances. User can update more than one shell at a time.

The user will be able to pass in a value for Location Picker element (`Datadefinition : uuu_location`) while updating the shell instance.

Location picker value will be validated.

User will be able to pass in a value for 'E-Signature Type' while updating the shell instance. 'E-Signature Type' value will be validated.

Support

This method only supports update of shells under a company and is not associated with any project.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Level	Yes or No
Program Level	No

Prototype

public XMLObject updateShell (String shortname, String authcode, String shellType, String shellXML);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
shellType	To identify type of the shell that is getting updated. Ex: Building, Projects, Capital Projects, Properties etc..,
shellXML	This will be based on Shell integration interface definition. Without this definition defined user will not be allowed to update a shell instance

shellXML Elements

Tag Name	Required	Description	Valid values
<_shell>	Yes	Base tag for new shell data. This tag can be repeated to create more than one shell instance	

Return Value

See Return Values. Message will contain Shell Number and Shell Name that is updated within Unifier.

Sample Method

updateShell("acme", "acme_authcode", "Buildings","<XML Content>");

Additional Information

Shell that will be updated will be based on shell type and shell number provided as part of XML content.

User will be allowed to update both CBS and Generic shells. For more information about shells refer to uDesigner Guide.

User will be able to pass in a value for Location Picker element while updating shell instance. Location picker value will be validated.

All other validation while creating shell instance manually will be honored through Integration viz.., required fields, duplicate shell id etc.

This method will allow user to update one more shell of same type at a time.

Get Shell List

Description

This method will allow users to get a list of shell instances. User can get information of more than one shell at a time.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project or Shell Level	No
Program Level	No

Prototype

public XMLObject getShellList(String shortname, String authcode, String shellType, String filterCondition);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
shellType	To identify type of the shell that is getting updated. Ex: Building, Projects, Capital Projects, Properties etc..,
filtercondition	This will allow user to filter list of shells returned. Following is list of elements that use can filter shell list on <shell number>: data element used to capture shell number <shell name>: data element used to capture shell name Status (uuu_shell_status): Status of the shell

Return Value

See ***Return Values*** (on page 645).

Note: The message will return list of shells based on filer condition.

Sample Method

getShellList("acme", "acme_authcode", "Buildings", uuu_shell_status=Active);

Additional Information

Only one filter condition can be provided at a time.

Only operator available for filter condition will be "=".

Example:

If user wants to get a list of shells with status Active then filterCondition should be <filterCondition>uuu_shell_status = Active</filterCondition>

If user wants to get a shell with name Building 101 then filterCondition should be <filterCondition>my_building_name = Building 101</filterCondition). Assuming that my_building_name is the data element used to capture name of the shell using SYS Shell Name.

Get Project/Shell List

Description

This method will allow users to get a list of project and shell instances. User can get information of more than one project and shell at a time.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project or Shell Level	No
Program Level	No

Prototype

public XMLObject getProjectShellList(String shortname, String authcode, String options);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication code for the company, in text string

Parameter	Description

Options	This parameter allows the user to filter the list based on the following:
	▸ <status>: select a status of Active, Inactive, On-Hold or View-Only. If no status is provided then all projects and shells are returned.
	▸ <type>: select the type to be returned. Value can be project, CBS_shell or generic_shell. If no type if provided then all projects and shells will be returned.
	▸ <filter_condition>: filter the results based on either the projectnumber or projectname.
	Example:
	`<gen:options>`
	`<![CDATA[`
	`<options>`
	`<type>CBS_shell</type>`
	`<status>Active</status>`
	`<filter_condition>projectname=Building 101</filter_condition>`
	`</options>`
	`]]>`
	`</gen:options>`

Return Value

See ***Return Values*** (on page 645).

> **Note**: The message will return list of project and shells based on filter conditions.

Sample Method

getShellList("acme", "acme_authcode", type=CBS_shell);

Schedule Manager Methods

In This Section

Create Schedule Sheet Activities from Primavera P5 and P6 XML 57
Create Schedule Sheet Activities V2 from Oracle® Primavera P6™ XML 58
Create Schedule Activities from file V2 60
Update Schedule Sheet Activities from Oracle® Primavera™ P5 and P6 XML 63
Update Schedule Sheet Activities V2 from Oracle® Primavera™ P6 XML 64
Update Schedule Activities From File V2 67
Get Schedule Sheet Activities from Unifier 70
Get a List of Project Schedule Sheets 71
Get a List of Schedule Sheet Data Mappings 72

Create Schedule Sheet Activities from Primavera P5 and P6 XML

Description

This method creates activities into an existing Unifier project or shell (cost code type CBS) level schedule sheet using default data mapping.

Support

This method can only support creation of activities in project or shell (cost code type CBS) level schedule sheets.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject createScheduleActivities(String shortname, String authcode, String projectNumber, String scheduleSheet, String scheduleXML);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string

Parameter	Description
projectNumber	Identifier of project or shell in Unifier.
scheduleSheet	Name of Schedule Sheet to import activities into.
scheduleXML	Content of activities to import. This is obtained from the Primavera P5/P6 xml content. The Primavera XML content should be generated by exporting the full Primavera project.

Return Value

See Return Values

Sample Method

createScheduleActivities("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XML Content>");

Additional Information

You must perform appropriate data mapping setups on the target schedule sheet. This includes mapping Unifier Schedule Sheet columns to Primavera XML elements, and setting appropriate XML Import options. Only elements that are mapped will be copied.

This method can only be used to create new activities not for updating existing activities.

If invoked on an empty schedule sheet, this method will insert activities into the schedule sheet. If invoked on a non-empty schedule sheet, it will overwrite all existing activities (delete and recreate) in the non-empty schedule sheet if this function is allowed in the data mapping setup.

If the method is called on a non-empty schedule sheet and the data mapping setup does not allow overwrite, then the method will return an error in the response.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

Activity identified with this creation method will be GUID.

Data Mapping Marked as default will be used.

Create Schedule Sheet Activities V2 from Oracle® Primavera P6™ XML

Description

This method creates activities into an existing Unifier project or shell (cost code type CBS) level schedule sheet based on the data mapping parameter and schedule options provided as part of Web Service call.

Support

This method can only support creation of activities in project or shell (cost code type CBS) level schedule sheets.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject createScheduleActivitiesV2(String shortname, String authcode, String projectNumber, String sheetName, String sheetXML, String scheduleOptions);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication code for the company, in text string
projectNumber	Identifier of project or shell in Unifier.
sheetName	Name of Schedule Sheet to import activities into.
sheetXML	Content of activities to import. This is obtained from the Primavera P5/P6 xml content. The Primavera XML content should be generated by exporting the full Primavera project.
scheduleOptions	This element will allow message to carry optional information. Following are the tags under this element ▸ <MapName/> ▸ <ActivityIdentifier/> ▸ <CBS_NoOfLevelsToMerge/> ▸ <CBS_StartMergeFrom/>
MapName	Name of the data mapping that should be used while processing XML file. If there is no map name then the one marked "Default" in Unifier on the schedule sheet will be used.
ActivityIdentifier	Possible values are GUID, ID. ID is the tag of Activity Id in Primavera. This tag will be used if you want to update an activity through udpateScheduleActivities webservice call. If Id tag is used for create and update then the value provided under Id tag will be used as task / activity identifier.
CBS_NoOfLevelsToMerge	This parameter will take a numeric value. This parameter can be used by user to specify how many levels (CBS Levels) in P6 XML file should be merged together. EXAMPLE: Assume that P6 has following structure: (5.ON.DA.DA03.F350.U11.095.ABC20.CAct04). If you specify 3, then three levels will be merged together and the 3 segments will be based on the next parameter (CBS_startmergefrom). Value of this parameter can be 0.

Parameter	Description
CBS_StartMergeFrom	This parameter will take a numeric value. Value of this parameter will be used to determine the level from which this merge should start. EXAMPLE: In the above example, if a user specifies 2, then system will start from level 2 and merge ON, DA and DA03. So after the merge the CBS Code build by import program will be 5.ONDADA03.F350.U11.095.ABC20.CAct04. Value of this parameter can be 0.

Return Value

See Return Values.

Sample Method

createScheduleActivitiesV2("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XML Content>","<scheduleoptions>");

Additional Information

You must perform appropriate data mapping setups on the target schedule sheet. This includes mapping Unifier Schedule Sheet columns to Primavera XML elements, and setting appropriate XML Import options. Only elements that are mapped will be copied.

This method can only be used to create new activities not for updating existing activities.

If invoked on an empty schedule sheet, this method will insert activities into the schedule sheet. If invoked on a non-empty schedule sheet, it will overwrite all existing activities (delete and recreate) in the non-empty schedule sheet if this function is allowed in the data mapping setup.

If the method is called on a non-empty schedule sheet and the data mapping setup does not allow overwrite, then the method will return an error in the response.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

- Activity identified with this creation method can be GUID or ID
- CBS Code will be derived automatically based on the XML file. For more information see Associate CBS Codes with Activities.

Create Schedule Activities from file V2

Description

This option will allow user to process a P6 XML file and add activities and resources to a Unifier schedule sheet. Along with activity information this process will also take activity spread information. Since the activity spread information data can be a lot, this method allows user to send in information in a .zip file.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject createScheduleActivitiesFromFileV2(String shortname, String authcode, String projectNumber, String sheetname, String scheduleoptions, String iszipfile, FileObject[] files);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
sheetName	Identify the name of the schedule sheet in Unifier
iszipfile	This tag identifies if the file provided under files tag is a zip file or .xml file with just activity and resource information. If user wants to send activity spread information then the value of this tag should be yes and the file provided under files tag should be a zip file containing activity/ resource information and the spread information.
files	FileObject containing the Activity / Resource information and the activity spread information. FileObject is made up of two components String filename (this should be the name of the zip file) and the zip file with two files, one for activity / resource information and one with activity spread information. Name of the activity/ resource information should be same as the one given under <scheduleoptions>.<ActivityFileName>tag and the name of the name of the activity spread information should be same as the one given under <scheduleoptions>.<SpreadFileName>.
scheduleOptions	This element will allow message to carry optional information. Following are the tags under this element <MapName/> <ActivityIdentifier/> <CBS_NoOfLevelsToMerge/> <CBS_StartMergeFrom/> <ActivityFileName/> <SpreadFileName/>

Parameter	Description
MapName	Name of the data mapping that should be used while processing XML file
ActivityIdentifier	Possible values are GUID, ID. ID is the tag of Activity Id in Primavera.
CBS_NoOfLevelsToMerge	This parameter will take a numeric value. This parameter can be used by user to specify how many levels (CBS Levels) in P6 XML file should be merged together. EXAMPLE: Assume that P6 has following structure: (5.ON.DA.DA03.F350.U11.095.ABC20.CAct04). If you specify 3 then three levels will be merged together, and the 3 segments will be based on the next parameter (CBS_startmergefrom)
CBS_StartMergeFrom	▸ This parameter will take a numeric value. Value of this parameter will be used to determine the level from which this merge should start. ▸ EXAMPLE: In the above example, if a user specifies 2, then system will start from level 2 and merge ON, DA and DA03. So after the merge the CBS Code build by import program will be 5.ONDADA03.F350.U11.095.ABC20.CAct04
ActivityFileName	Name of the file that contains the activity and resource information. Please refer to files tag
SpreadFileName	Name of the activity spread file name. Please refer to files tag

Return Value

See Return Values.

Sample Method

createScheduleActivitiesFromFileV2("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XMLContent>","yes","<scheduleoptions>",files);

Additional Information

Format of spread file should be similar to Sample Spread Data example.

While processing spread information, Unifier will only look for <amt> tag.

If the Import from external source option is checked in Unifier and spread information is not available in XML file then system will error out. This error condition is also valid when this tag is not available in the XML file.

Activity File should start and end with <List_Wrapper> tag

Spread file should not contain any <List_Wrapper> tag

Update Schedule Sheet Activities from Oracle® Primavera™ P5 and P6 XML

Description

This method will insert/update activities into any project or shell schedule sheet. It will not overwrite existing activities in the schedule sheet.

Support

This method can only update activities in project or shell level schedule sheets.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project or Shell Level	Yes

Prototype

public XMLObject updateScheduleActivities(String shortname, String authcode, String projectNumber, String sheetName, String sheetXML);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
projectNumber	Identifier of project or shell in Unifier.
sheetName	Name of Schedule Sheet to import activities into.
sheetXML	Content of activities to import. This is obtained from the Primavera P5/P6 xml content. The Primavera XML content should be generated by exporting the full Primavera project.

Return Value

See Return Values.

Sample Method

updateScheduleActivities("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XML Content>");

Additional Information

You must perform appropriate data mapping setups on the target schedule sheet. This includes mapping Unifier Schedule Sheet columns to Primavera XML elements, and setting appropriate XML Import options.

This method will insert/update activities into any project or shell schedule sheet. It will not overwrite existing activities in the schedule sheet. The method will utilize the data mapping setup to update only those elements that are mapped.

The method errors out if the data mapping option does not allow merge.

Activities are uniquely identified within Primavera by the Globally Unique Id (<GUID>). The GUID is assumed to be unique not just within Primavera but across all products including Unifier.

Resource assignments will be imported (inserted/updated) if the 'Import Resource Assignments' checkbox is selected. Resource assignments will be matched by Resource Name.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

Update Schedule Sheet Activities V2 from Oracle® Primavera™ P6 XML

Description

This method will insert/update activities into any project or shell schedule sheet. It will not overwrite existing activities in the schedule sheet based on the data mapping parameter.

Support

This method can only support updation of activities in project or shell level schedule sheets.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project or Shell Level	Yes

Prototype

public XMLObject updateScheduleActivitiesV2(String shortname, String authcode, String projectNumber, String sheetName, String sheetXML, String scheduleOptions);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
projectNumber	Identifier of project or shell in Unifier.
sheetname	Name of Schedule Sheet to import activities into.
sheetXML	Content of activities to import. This is obtained from the Primavera P5/P6 xml content. The Primavera XML content should be generated by exporting the full Primavera project.

Parameter	Description
scheduleOptions	This element will allow message to carry optional information. Following are the tags under this element <MapName/> <ActivityIdentifier/> <CBS_NoOfLevelsToMerge/> <CBS_StartMergeFrom/> <ActivityDeletion/>
MapName	Name of the data mapping that should be used while processing XML file
ActivityIdentifier	Possible values are GUID, ID. ID is the tag of Activity Id in Primavera.
CBS_NoOfLevelsToMerge	This parameter will take a numeric value. This parameter can be used by user to specify how many levels (CBS Levels) in P6 XML file should be merged together. ▸ EXAMPLE: Assume that P6 has following structure (5.ON.DA.DA03.F350.U11.095.ABC20.CAct04) If you specify 3 then two levels will be merged together. But which 3 segments will be based on the next parameter (CBS_startmergefrom)
CBS_StartMergeFrom	▸ This parameter will take a numeric value. Value of this parameter will be used to determine the level from which this merge should start. ▸ EXAMPLE: In the above example, if a user specifies 2, then system will start from level 2 and merge ON, DA and DA03. So after the merge the CBS Code build by import program will be 5.ONDADA03.F350.U11.095.ABC20.CAct04
ActivityDeletion	▸ Indicate to delete activities from schedule sheet in Primavera Unifier if it is deleted in P6.

Return Value

See Return Values.

Sample Method

updateScheduleActivitiesV2("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XML Content>","Data Mapping 001","Id");

Additional Information

You must perform appropriate data mapping setups on the target schedule sheet. This includes mapping Unifier Schedule Sheet columns to Primavera XML elements, and setting appropriate XML Import options.

This method will insert/update activities into any project or shell schedule sheet. The method will utilize the data mapping setup to update only those elements that are mapped.

The method errors out if the data mapping option does not allow merge.

Activities are uniquely identified within Primavera by the Globally Unique Id (<GUID>) or the identified provided as part of XML file. GUID will be used if ActivityIdentifier is empty or blank.

Resource assignments will be imported (inserted/updated) if the 'Import Resource Assignments' checkbox is selected. Resource assignments will be matched by Resource Name.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

- Activity identified with this creation method can be GUID or ID
- On the options tab of Data Mapping, merge existing schedule option should be checked.
- CBS Code will be derived automatically based on the XML file. For more information see Associate CBS Codes with Activities.
- ActivityDeletion tag logic is based on the "Delete Activity" checkbox available on data mapping window in Unifier. Following is the logic for this tag

Delete Checkbox	Activity Deletion	Behavior
Unchecked	Auto or Confirm	Activity Deletion tag is ignored. Activities that are removed under source schedule will not be removed from Unifier Schedule Sheet
Checked	Auto	Activity Deletion tag is considered Activities that are removed under source schedule will be deleted automatically under Unifier Schedule Sheet.
Checked	Confirm	Activity Deletion tag is considered Activities that are removed under source schedule will not be deleted. System will return an error and will not update Primavera Unifier schedule sheet. If no activities are removed under source schedule then system will update Unifier Schedule Sheet.

ActivityDeletion tags are only considered for updateScheduleSheetFromFileV2 service and not for create.

Update Schedule Activities From File V2

Description

This option will allow user to process a P6 XML file and add and update activities and resources to a Unifier schedule sheet. Along with activity information this process will also take activity spread information.

Support

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

public XMLObject updateScheduleActivitiesFromFileV2(String shortname, String authcode, String projectNumber, String sheetname, String scheduleoptions, String iszipfile, FileObject[] files);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
sheetName	Identify the name of the schedule sheet in Primavera Unifier
iszipfile	This tag identifies if the file provided under files tag is a zip file or .xml file with just activity and resource information. If user wants to send activity spread information then the value of this tag should be yes and the file provided under files tag should be a zip file containing both activity and resource information and the spread information.

Parameter	Description
files	FileObject containing the Activity / Resource information and the activity spread information. FileObject is made up of two components String filename (this should be the name of the zip file) and the zip file with two files, one for activity / resource information and one with activity spread information. Name of the activity/ resource information should be same as the one given under <scheduleoptions>.<ActivityFileName>tag and the name of the name of the activity spread information should be same as the one given under <scheduleoptions>.<SpreadFileName>.
scheduleOptions	This element will allow message to carry optional information. Following are the tags under this element <MapName/> <ActivityIdentifier/> <CBS_NoOfLevelsToMerge/> <CBS_StartMergeFrom/> <ActivityDeletion> <ActivityFileName> <SpreadFileName>
MapName	Name of the data mapping that should be used while processing XML file
ActivityIdentifier	Possible values are GUID, ID. ID is the tag of Activity Id in Primavera.
CBS_NoOfLevelsToMerge	This parameter will take a numeric value. This parameter can be used by user to specify how many levels (CBS Levels) in P6 XML file should be merged together. ▸ EXAMPLE: Assume that P6 has following structure (5.ON.DA.DA03.F350.U11.095.ABC20.CAct04) If you specify 3 then two levels will be merged together. But which 3 segments will be based on the next parameter (CBS_startmergefrom)

Parameter	Description
CBS_StartMergeFrom	▸ This parameter will take a numeric value. Value of this parameter will be used to determine the level from which this merge should start. ▸ EXAMPLE: In the above example, if a user specifies 2, then system will start from level 2 and merge ON, DA and DA03. So after the merge the CBS Code build by import program will be 5.ONDADA03.F350.U11.095.ABC20.CAct04
ActivityDeletion	This tag will allow user to determine if the activities that does not exists in P6 should be deleted from Unifier schedule sheet. Possible values are Auto / Confirm.
ActivityFileName	Name of the file that contains the activity and resource information. Please refer to files tag
SpreadFileName	Name of the activity spread file name. Please refer to files tag

Return Value

See Return Values.

Sample Method

updateScheduleActivitiesFromFileV2 ("acme","acme_authcode","Proj-0001","Schedule Sheet 001", "<XML Content>","yes","<scheduleoptions>",files);

Additional Information

ActivityDeletion tag logic is based on the "Delete Activity" checkbox available on data mapping window in Unifier. Following is the logic for this tag

Delete Checkbox	Activity Deletion	Behavior
Unchecked	Auto or Confirm	Activity Deletion tag is ignored. Activities that are removed under source schedule will not be removed from Unifier Schedule Sheet
Checked	Auto	Activity Deletion tag is considered Activities that are removed under source schedule will be deleted automatically under Unifier Schedule Sheet.

Delete Checkbox	Activity Deletion	Behavior
Checked	Confirm	Activity Deletion tag is considered Activities that are removed under source schedule will not be deleted. System will return an error and will not update Primavera Unifier schedule sheet. If no activities are removed under source schedule then system will update Unifier Schedule Sheet.

ActivityDeletion tags are only considered for updateScheduleSheetFromFileV2 service and not for create.

While processing spread information, Unifier will only look for "" tag.

If the Import from external source option is checked in Unifier and spread information is not available in XML file then system will error out. This error condition is also valid when this tag is not available in the XML file.

Activity File should start and end with <List_Wrapper> tag

Spread file should not contain any <List_Wrapper> tag

Get Schedule Sheet Activities from Unifier

Description

This method will extract activities from any project or shell (cost code of type CBS) schedule sheet.

Support

This method can only extract activities from project or shell level schedule sheets.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject getScheduleActivities(String shortname, String authcode, String projectNumber, String scheduleSheet);

Parameters

Parameter	Description
Shortname	Identifier of the company, company's short name
Authcode	Authentication code for the company, in text string
projectNumber	Identifier of project or shell in Unifier.
scheduleSheet	Name of Schedule Sheet to import activities into.

Return Value

See Return Values.

Sample Method

getScheduleActivities("acme", "acme_authcode", "Proj-0001","Schedule Sheet 001");

Additional Information

The XML return object contains a Unifier specific format. You will need to write additional integration logic to convert it to other formats.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

Get a List of Project Schedule Sheets

Description

This method will return a list of Schedule Sheets within a Project or Shell.

Support

This method can only list Schedule Sheets from projects or shells.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject getScheduleSheetList(String shortname, String authcode, String projectnumber, String options);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
projectnumber	Identifier of project or shell in Unifier.
options	This parameter allows the user to filter the list based on the following: <status>: select a status of Active or Inactive. If no status is provided then all sheets are returned. <main_sheet>: Set to true or false in order to specifiy if the master sheet is to only be returned. Setting true will return only the master schedule. <filter_condition>: two filter conditions are available; name and sheet_lock. The name is the name of the sheet and sheet_lock can have a value of true or false in order to only return locked schedule sheets.

Return Value

See Return Values.. Message will return list of Schedule Sheets based on filter condition

Sample Method

getScheduleSheetList("acme", "acme_authcode", "P-000001", status=Active);

Get a List of Schedule Sheet Data Mappings

Description

This method will return a list of Data Mappings configured for a Schedule Sheet within a Project or Shell.

Support

This method can only list Data Mappings from Schedule Sheets within projects or shells.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject getScheduleSheetDataMaps(String shortname, String authcode, String projectnumber, String sheetname, String options);

Parameters

Parameter	Description
shortname	Identifier of the company, company's short name
authcode	Authentication code for the company, in text string
projectnumber	Identifier of project or shell in Unifier.
sheetname	Identifier of the schedule sheet within the project or shell in Unifier.
options	This parameter allows the user to filter the list based on the following: <datamap_name>: The name of a specific data mapping to be retrieved. <filter_condition>: one filter conditions is available; isdefault. If set to true only the data mapping flagged as the default will be returned.

Return Value

See Return Values. Message will return list of Schedule Sheets based on filter condition

Sample Method

getScheduleSheetDataMaps ("acme", "acme_authcode", "P-00001", "Master Schedule", isdefault=true);

Asset Manager Methods

In This Section

Create Asset 75

Create Asset

Description

This method creates assets under an asset class.

Support

This method can only support creation of assets at company level.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	No
Program Level	No

Prototype

public XMLObject createAsset(String shortname, String authcode, string assetClassName, String copyFromAsset, String assetXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
assetClassName	Name of asset class (Buildings, IT Equipments etc..,)
copyFromAsset	This can be another asset code or a template asset code. User can provide a null value if user does not want to copy from an asset or template. If user provides a value, system should first look under templates and if template does not exists then it should check under real assets
assetXML	Content of the asset that needs to be created. This is based on Integration interface design for asset class in uDesigner.

Return Value

See Return Values.

Sample Method

createAsset("acme", "acme_authcode", "Buildings","Buildings Temp 1","XML")

Additional Information

Same logic while creating new assets from UI will be used while creating an asset from integration. This includes validation logic, setting status of an asset, depreciation definitions etc..,

If user tries to create an asset without copying from another asset or template, then new asset will be created based on the data provided as part of web services payload. Note: No depreciation setup will be available for these newly created assets as payload does **not take any depreciation setup information.**

If user tries to create an asset by copying from another asset or template then new asset will be created based on the data provided as part of web services payload. Template or asset name provided as part of web services call will be used to copy depreciation setup only (depreciation data will not be copied). That means only deprecation information will be copied from the template or asset and not any data on the detail form of template.

All validations on asset detail form including required fields and user defined form validations will be honored while creating new assets through web services.

If depreciation is available for newly created assets then the depreciation schedule will be calculated, asset detail form will be updated as part of web services call.

Once asset is created, asset detail form will be in Finish Editing mode and data including depreciation information will be rolled up to asset class sheets, asset summary sheet and accounts sheet.

Data Element which is based on data definition viz.., SYS Numeric Logical Datasource, SYS Date Logical Datasource, SYS Business Process Datasource and SYS Project Cost Datasource will not be available as part of Integration.

Following data element are only available as part of createAsset

Asset Code and Asset Navigation Code: These two data elements are built automatically based on the definition

Net Book Value, Cumulative Depreciation, Current Period Depreciation and Calculations as of: Calculated based on asset depreciation setup

User Defined Reports (UDR) Methods

In This Section

Get UDR Data 77

Get UDR Data

Description

This method gets XML data from UDR output.

Support

This process is supported under both Company and Project or Shell level reports. Only reports that are marked for Integration are eligible for integration.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject getUDRData(String shortname, String authcode, string projectNumber, String reportName);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or shell in Unifier. If you want to get company level report then pass null for this parameter.
reportName	name of the report user defined report

Return Value

Response will contain following data elements

Tag Name	Description
<report>	Main report tag

Tag Name	Description
<report_header>	Element that carries all report header column names.
<report_row>	Element that carries data of all rows. This element will also show following information **Group By**: If report has group by condition then the data value will show under this element **Subtotal:** If report is configured to show sub-total then the data value will show under this element **Count:** If report is configured to show count then the data value will show under this element.

Sample Method

getUDRData ("acme", "acme_authcode", "project_info", "Funding Report")

The XML output of an UDR will be based on report design.

- Tabular UDR with no Group
- Tabular UDR with Group
- Tabular UDR with Sub-Total
- Tabular UDR with Count
- Cross Tab UDR
- Summary UDR
- Alert UDR

Additional Information

A user-defined report (UDR) always runs upon receiving the message. This ensures that the latest data is returned.

Web Services call does not need any additional permission to run a UDR that is marked for Integration

If UDR has query parameters defined then the Web Services method will honor that as long as the query parameters are pre-populated during report design

Following are the failure reasons and description

Reason	Description
Report name is not valid. Check if report exists or is enabled for Integration	Invalid Report Name (If report does not exist or is not marked for Integration)
Unable to run report. Contact System Administrator	While running report if there is a system error or for any reason Unifier cannot run this report

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

In case of the getUDR web-service call:

- If the report data being retrieved is based on permission aware data sources, then nothing will be returned.
- If the report being retrieved is based on system defined data sources, then it will work as per existing functionality.

Cost Sheet Methods

In This Section

Get Column Data..... 81
Update Column Data..... 83

Get Column Data

Description

This method gets data from a project or shells (with cost codes CBS) cost sheet column. The getColumnData does not support Assigned Budget column although it is a manual column.

Support

This method can only support for project or shell level cost sheet column that is defined as Manual Entry Method (either Direct Entry or Line Item Content)

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject getColumnData(String shortname, String authcode, string projectNumber, String ColumnName);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
columnName	Identifier of the cost sheet column in Unifier. This is case sensitive, and should be the name of Column that is defined in Unifier.

Return Value

See Return Values. Response will contain following data elements.

Tag Name	Description
<CBS_code>	CBS Code is a required field.
<short_description>	Short Description
<long_description>	Long Description
<work_package>	Work Package Name
<spends_category>	Spends Category Label
<quantity>	Quantity
<unit_of_measure>	Unit of Measure
<unit_cost>	Unit Cost Label
<amount>	Amount

Sample Method

getColumnData("acme", "acme_authcode", "project_info","my column name")

Additional Information

Following are the failure reasons and description

Reason	Description
Column is not valid	Invalid Column Name
Column is not manual	Column is not defined as manual entry

If the cost sheet column entry method is "Line Item Content" then this web service call will only return lines for CBS Code which has valid lines. For example, if there are 2 CBS Codes on cost sheet and one has line item content and other does not then only CBS Code with line item content will be returned.

If the cost sheet column entry method is "Direct Entry" then this web service call will return a line for a CBS Code even if there is no data manually entered against that CBS Code in project or shell cost sheet. For example, if there are 2 CBS Codes on cost sheet and one has a value entered directly on cost sheet and other does not have any value, even then this service call will return 2 lines, one for each CBS Code. Following are the elements that will be returned as part of each CBS Code.

Observe that short_description element will automatically get a value "Direct Entry Line Item". This is to indicate that the value was entered directly on cost sheet.

This service can be used on Project (Standard) and Shells of cost code type CBS.

Update Column Data

Description

This method updates data of a project or shells (cost code type CBS) cost sheet column. This method replaces entire data.

Support

This method can only support for project or shell level cost sheet column that is defined as Manual Entry Method. This column does not support Assigned Budget column. This method does not support incremental or delta changes to a column.

Installation: ASP and Self host

Level	Yes or No
Company Level	No
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject updateColumnData(String shortname, String authcode, string projectNumber, String ColumnName, String DataXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
columnName	Identifier of the cost sheet column in Unifier. This is case sensitive, and should be the name of Column that is defined in Unifier.
columnXML	identifier for Column content in XML format

Return Value

DataXML Elements

The elements that include as part of DataXML parameter are:

Tag Name	Description
<CBS_code>	CBS Code is a required field.
<short_description>	This is a required field.
<long_description>	This is an optional field

Tag Name	Description
<work_package>	This is an optional field. But use should specify a work package name which is valid under a project or shell. Data entered under this element will be ignored for company level cost column
<spends_category>	This is an optional field. Data sent through this data element should be validated against the data set values
<quantity>	This is an optional field. Data sent through this data element should be numeric. Any other data will be considered as illegal and error should be returned back to user. If user did not send a value for this then "0" will be assumed as value.
<unit_of_measure>	This is an optional field. Data sent through this data element should be validated against the data set values
<unit_cost>	This is an optional field. Data sent through this data element should be numeric. Any other data will be considered as illegal. If user did not send a value for this then "0" will be assumed as value.
<amount>	This is an optional field. User cannot enter any value for this element. Value of this data element should be automatically calculated as (Quantity * Unit Cost)

Project or Shell → Administrator will be used as user who is performing this integration. Audit log will track and trace all changes done through this web services method.

Sample Method

updateColumnData("acme", "acme_authcode", "project_info","my column name", "DataXML")

Additional Information

Following are the failure reasons and description

Reason	Description
Column is not valid	Invalid Column Name
Column is not manual	Column is not defined as manual entry
Administrator is inactivated	Administrator is inactive. Cannot create column data.

This service can be used for both Project (Standard) and Shells of cost code type CBS.

Project Methods

In This Section

Create Project 85
Create Project with additional options..... 87

Create Project

Description

This method creates a project in Unifier either based on a template or an existing project.

Support

This method can only support creation of project.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Not Applicable

Prototype

public XMLObject createProject(String shortname, String authcode, String copyFromProject, String projectXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
copyFromProject	this can be a project number or a template number.
projectXML	content of the project that needs to be created.

projectXML Elements

Tag Name	Required	Description	Valid values
<project_info>	Yes	Base tag for new project data. Only one project can be created at a time.	

Tag Name	Required	Description	Valid values
<projectnumber>	Yes	Project number of the new project that is getting created. If this project number already exists you will get an error.	User defined number.
<projectname>	Yes	Project name of the new project that is getting created. If this project name already exists you will get an error.	User defined name.
<constructiontype>	No	Type of construction (New Construction or Retrofit/Remodel).	New Construction: 1 Retrofit/Remodel: 0
<status>	No	Status of the Project (Active, Inactive, On-Hold)	Active: 1 Inactive: 0 On-Hold: 2
<typeofproject>	No	Type of Project. Values are based on the user dataset for "Project Type" data definition	
<projectsite>	No	Project site. Values are based on the user dataset for "Project Site" data definition	
<startdate>	No	Start Date of the project.	
<plannedcompletion>	No	Planned Completion date	
<revisedcompletion>	No	Revised Completion date	
<designcomplete>	No	Design % complete	
<constrcomplete>	No	Construction % complete	
<notes>	No	Notes	
<schedulestatus>	No	Schedule status of the project	On Schedule and on Budget: 0 Potential impact on schedule or both: 1 Impact to schedule or Budget or both: 2

Tag Name	Required	Description	Valid values
<projectphase>	No	Project phase information. Values are based on the user dataset for "Project Phase" data definition	

Note: If you do not provide values for elements that are not required then the values from template or project that is used to create new project are used.

Return Value

See Return Values. Message will contain new Project number and name that is created within Unifier.

Sample Method

createProject ("acme", "acme_authcode", "Proj-001", "<XML Content>")

Additional Information

If you are sending a message to update user information then you have to send a user name that already exists in Unifier.

If a new project is created by copying from a Template or another existing Project then users and groups will be copied by default along with permissions.

This service can be used to create Project (Standard).

Create Project with additional options

Description

This method creates a project in Unifier either based on a template or an existing project. This method is enhanced to have additional options to support copying of modules and address information from template or project.

Support

This method can only support creation of project.

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project Level	Not Applicable

Prototype

public XMLObject createProject(String shortname, String authcode, String copyFromProject, String projectXML);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
copyFromProject	this can be another project number or a template number.
projectXML	content of the project that needs to be created.

projectXML Elements

Tag Name	Required	Description	Valid values
<project_info>	Yes	Base tag for new project data. Only one project can be created at a time.	
<projectnumber>	Yes	Project number of the new project that is getting created. If this project number already exists you will get an error.	User defined number.
<projectname>	Yes	Project name of the new project that is getting created. If this project name already exists you will get an error.	User defined name.
<constructiontype>	No	Type of construction (New Construction or Retrofit/Remodel).	New Construction: 1 Retrofit/Remodel: 0
<status>	No	Status of the Project (Active, Inactive, On-Hold)	Active: 1 Inactive: 0 On-Hold: 2
<typeofproject>	No	Type of Project. Values are based on the user dataset for "Project Type" data definition	
<projectsite>	No	Project site. Values are based on the user dataset for "Project Site" data definition	

Tag Name	Required	Description	Valid values
<startdate>	No	Start Date of the project.	
<plannedcompletion>	No	Planned Completion date	
<revisedcompletion>	No	Revised Completion date	
<designcomplete>	No	Design % complete	
<constrcomplete>	No	Construction % complete	
<notes>	No	Notes	
<schedulestatus>	No	Schedule status of the project	On Schedule and on Budget: 0 Potential impact on schedule or both: 1 Impact to schedule or Budget or both: 2
<projectphase>	No	Project phase information. Values are based on the user dataset for "Project Phase" data definition	
<description>	No	Description of the project	
<location>	No	Base tag for project location information	This is a new tag added for this enhancement. This tag is optional. If this tag is not provided then Primavera Unifier will create Project by copying "Project Address" information from template.
<address>	No	Base tag for address information	This tag can be repeated to provide multiple address types
<address1>	Yes	Address 1 field of the project	Is child element of <address> element. This element is required if address tag is provided
<address2>	No	Address 2 field of the project	Is child element of <address> element

Tag Name	Required	Description	Valid values
<city>	Yes	City field of the project	Is child element of <address> element. This element is required if address tag is provided
<state>	Yes	State field of the project	Is child element of <address> element. This element is required if address tag is provided
<country>	Yes	Country field of the project	Is the child element of <address> element. This element is required if the address tag is provided. The value comes from the Return Values > Country. The User must send in Label as a valid value
<phone>	No	Phone field of the project	Is child element of <address> element
<fax>	No	Fax filed of the project	Is child element of <address> element
<addresstype>	Yes	Type of address	Is child element of <address> element Following are the valid values for this element ▸ Project Address ▸ Billing ▸ Shipping ▸ Billing and Shipping
<options>	No	Base tag for options.	This is a new tag added for this enhancement. This tag is optional. If this tag is not provided then Primavera Unifier will create Project by copying user / groups from another project or template.

Tag Name	Required	Description	Valid values
<copyModules>	No	Option to either copy modules or not.	This tag can only go as a child element under <options> tag. This tag is identify either to create a new project by copying all modules including user / group or just creates a new project by copying user / group from another project or template. Only valid values are Yes / No. Yes means copy all modules **including user / groups. No means copy only user / groups. If tag is not provided or value is empty then default behavior will be to create project by copying user / groups.**
<projectCurrency>	No	Option to specify the project currency	A new child element of <projectCurrency> will be available within the existing <options> tag. Valid values for this tag are the code for the currency rather than the names. For example, USD is valid, United States Dollars is not. If the code is not a valid code then the web service will fail. If the tag is provided and the value is empty then template code will be selected.

Note: If you do not provide values for elements that are not required then the values from template will be used to create new project.

Return Value

See Return Values. Message will contain new Project number and name that is created within Unifier.

Sample Method

createProject ("acme", "acme_authcode", "Proj-001", "<XML Content>")

Additional Information

If a new project is created by copying from a Template or another existing Project then users and groups will be copied by default along with permissions.

Address information provided through XML tags is of type Project Address, Billing, Shipping and Billing and Shipping.

For a list of valid country names, see Currency.

If a company is configured to have auto numbering for projects then any information send through integration for projectnumber tag will be ignored and system will provide a number automatically based on sequence.

This service can be used create Project (Standard).

Business Process (BP) Methods

In This Section

Create BP Record 93
Create Complete BP Record 101
Add BP Line Item 106
Add Complete BP Line Item 109
Update BP Record 114
Update Complete BP Record 115
Update BP Record V2 118
Get BP Record 126
Get BP List 128
Get Complete BP Record 131

Create BP Record

Description

This method can be used to:

- Create new business process records in Unifier based on Integration design that has been defined in uDesigner. For more information refer to *uDesigner Administrator Guide*.
- Create group line items together. You can assign a group name to provide the context of the line items.

If you want to create a BP record with line item information, then you must use this service. If this is a cost related BP, line items will be rolled up to the cost sheet.

> **Note**: When you a create record through a web service, and you want to send the newly created record with a record number (record_no) value to Unifier, to avoid duplicating record numbers, do not include a value for the next record number because the value has already been configured in the BP setup. You can only enter a value in the record number (record_no) field, if you are getting the value from an external system, or you are self-generating a value, and the value does not fall into the same sequence number pattern as set in the BP setup. In this scenario, when you are sending the record number (record_no) value to Unifier, it is important to know that providing a unique record number (record_no) value is your responsibility.

Support

This process supports all type of BPs

Installation: ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject createBPRecord(String shortname, String authcode, string projectNumber, String BPName, String BPXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier. If you want to create a BP record at Company level then pass null for this parameter.
BPName	Identifier of the business process in Unifier, for example: Invoice. This is case sensitive, and should be the name as given in uDesigner
BPXML	identifier for BP content in XML format

BPXML Elements

The elements that include as part of BPXML parameters is dependent on the Integration interface design of your BP in uDesigner. XML format can be exported (downloaded) from uDesigner. Elements with direction marked as "Input" or "Both" will be part of XML format.

Note that elements lists should be send based on the type of BP and your design in uDesigner.

Following a list of elements that can be included as part of integration design in uDesigner and behavior when you include them. Note that elements lists should be sent based on the type of BP for which you are creating a record.

Tag Name	BP Type	Description
<amount>	Cost	Amount is a required field if you are sending line item information under _bp_lineitems. You will see same value on Unifier.
<CBS_code>	Cost	CBS Code is a required field if you are sending line information under _bp_lineitems.

Tag Name	BP Type	Description
<status>	Any	a) If BP uses a status field then you can send status tag. b) If the BP is a workflow based then the record will be created and will stay in the initial status that is valid. User has to go through remaining workflow process. c) If it is non-workflow based then the message can send a valid status and the record will take that status. If there is only one status defined for the BP then by default that status will be picked even if user sends a wrong status value.
<record_no>	Any	You can send record number by using this tag. If you do not specify a record_no tag as part of message then a new record will be created. Unifier will assign a record number. If you specify a record_no then system will create a record with the given record number. If the record number is a duplicate number then an error message will be send back as response. If the record_no is specified and if it does not exists then a new record will be created.
<_comment>	Text BP	Enter the responses (comments) associated with the BP record.
<li_group_name>	All BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.	Use to specify group name.

Return Value

See Return Values. This is dependent on which WSDL you are going to use.

If you use WSDL –1 then you should expect XMLObject as return value.

If you use WSDL – 2 then you should expect XMLFileObject as return value.

Create BP Record (Additional Information)

This topic includes additional information about creating BP records.

BP XML Structure

To get the BP XML structure, first send a message to get BP record. Refer to the Web Service getBPRecord for more information.

Interface

This interface can be extended with additional data elements. In order for Unifier to recognize those elements, the data element name must match those created in uDesigner for this business process.

YTB/AFC

If the tags YTB or AFC are present in the line items, Unifier processes these tags. If they are not present, Unifier will check the uDesigner BP setup configuration for Cost Adjustment. If this field has been set, the line item amount is treated as YTB/AFC and processed.

Single Record BP

Record Exists

If you send a message for single record BP and a record already exists then Unifier will not send back and error but if you send an invalid record number then Unifier will send back an error.

Record does not exist

If you send a message for single record BP and a record does not exist then Primavera Unifier will create a record for you. If the BP type is of Line item then any line items included in the message will be added to the BP record. If the BP type is of Simple then any line items added as part of message will be ignored.

If your BP has **uuu_creation_date** then it will be populated when you send a message

If your Cost BP has fund related information then you have to send Fund code details. If you are using multi segment fund code then send the code segments with delimiters.

Example:

Fundseg1-Fundseg2-Fundseg3.

The fund information send as part of Cost BP will be rolled up to Funding Sheet or Cost Sheet.

You cannot send fund related information for Cost Type BPs such as General Spends and Payment Applications.

All data element values provided as part of integration message will be validated against the action form selected as part of design in uDesigner. Validation includes required field, form validation etc..,

The following XML tag <_refnum> </refnum> should be added under BP line item tag <_bp_lineitem> </bp_lineitem> to support the ability to import the line items for General Spends and Payment Applications if SOV is individual line items. For Group by commit codes lines are identified by CBS Code.

If the action form used for validation is auto-populating values from a Business Process picker then system will auto-populate these values if a valid value is provided for that Business Process Picker.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Additional element and sub elements (XML tags) within the line items (_bp_lineitems) of Summary Payment Application SOV Type

To perform Cost allocation to line items, use the following information:

Note: For additional details, refer to the Unifier Reference Guide.

Tag Name	BP Type	Description
<_cost_allocation>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	Cost allocation for the summary line item will be done using _cost_allocation.
<bItemID>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	bItemID, which is CBS Code, is a required field if you are sending costed line item information under _cost_allocation.
<short_desc>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	Required field.
<uuu_quantity>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	Required field if cost line item type specified in _bp_lineitems has been set as Unit Cost. The Unit Cost information is specified in _bp_lineitems.
<amount>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	Required field if cost line item type specified in _bp_lineitems has been set as Lump Sum.

Tag Name	BP Type	Description
<_bp_lineitems>	Cost (Base Commit and Change Commit of Summary Payment Application SOV type)	Optional field. Use when modifications are needed for line items that have already been committed in an SOV.

The Payment Application detail form is a customized form that is designed in uDesigner. This form will have a set of fixed and user-defined fields. The Input XML for Payment Application will include the following:

- All existing XML tags that are currently applicable for the classic Payment Application Business Process.
- The tags listed in the following table:

Note: The table only lists out the system defined Data Element behavior specific to Payment Applications of SPA SOV type

Available for data input in Elements	Tags / Data elements specific to Payment Application	Additional comments
_bp_lineitems and _cost_allocation	Reference (_refnum)	This will be the identifier for the line item that needs to be paid. This field cannot be empty when the _bp_lineitems block has values in other fields. This will be the identifier for the costed line item as well.
_cost_allocation	bItemID (Cost Code)	This field cannot be empty when the _cost_allocation block has values in the other fields. Combination of _refnum and bItemID will be used to identify the costed line against which the payment has to be made. If there are duplicate _refnum and CBS codes in multiple cost allocation blocks then the creation/update will fail.
_bp_lineitems and _cost_allocation	short_desc (Description)	Both at summary and costed line levels.
_bp_lineitems and _cost_allocation	uuu_spa_qty_tp (Quantity this Period)	Since the Input XML will be common for both Lump Sum and Unit cost the tags will always be available. Any value entered in this field when cost type is Lump Sum, will be ignored.
_bp_lineitems and _cost_allocation	uuu_spa_amt_tp (Amount this Period)	Since the Input XML will be common for both Lump Sum and Unit cost the tags will always be available. Any value entered in this field when cost type is Unit Cost, will be ignored.

Available for data input in Elements	Tags / Data elements specific to Payment Application	Additional comments
_bp_lineitems and _cost_allocation	uuu_spa_other_tp (Other Amount this Period	Both summary line items and costed lines can have this field value.
_bp_lineitems and _cost_allocation	uuu_spa_mat_stored (Material Stored)	Both summary line items and costed lines can have this field value.
_bp_lineitems	uuu_spa_per_comp (Percentage Complete To Date)	Only summary line items can have this field value.

Notes:

- The "_bp_lineitems" elements will be as per the detailed integration interface design.
- The "_cost_allocation" elements will mostly be the same as "_bp_lineitems" elements, but the editability of the fields is governed by the logic currently existing in the system.
- The Payment Application has certain restrictions regarding fields that can be edited at the Summary or Allocation levels:
- If the XML template has values in the "_bp_lineitems" elements that are not allowed, for example, CBS codes, then such values are ignored.
- There are certain fields such as "uuu_cost_li_type" which should be entered only at the Summary level.
- If the XML has the value in the "_cost_allocation", then such values are ignored.

Behavior Specific to Summary Payment Application SOV type BPs

Note: The following information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV type.

- Business Process records can be created both with and without line items, by using various Create methods.
- Standalone Create methods also exist by using a line items that can be added to an existing Business Process record, for example, Add BP Line Item.

In both cases mentioned above, you can add the summary line items along with the cost allocation.

For adding line items to the Change Commit BP records from SOVs (modifying committed line items) using Input XMLs use the following:

- The reference tag uuu_sovlinum in the _bp_lineitems element determines the SOV line that will be modified. The costed lines that need to be updated will be identified by the uuu_sovlinum and Cost Code/ bItemId. The combination of uuu_sovlinum and Cost Code/ bItemId must be unique. Post record creation of all costed lines belonging to the summary line item will be added.
- If the input XML has the cost line item type field, then the system checks to ensure that the uuu_sovlinum and uuu_cost_li_type are as per SOV line. If there is a mismatch, the system generates an integration error.
- If the input XML has both uuu_sovlinum and uuu_cost_li_type fields, but the uuu_cost_li_type field has an invalid value, then the system generates an integration error.
- If the input XML has just the uuu_sovlinum and the uuu_cost_li_type field is blank, then no check will be performed for cost line item type validation.
- The Short description from a committed line comes from the SOV. When a line item has the uuu_sovlinum value, the short description field can empty. At runtime the value comes from SOV; however, if the short description value is included in the XML, then the existing value replaces the one coming from SOV.

Payment Application

The following information is specific to Payment Application.

At the time of creating a Payment Application record, details of the line items that are getting paid will be entered in the Input XML. Post creation of the record, all the line items existing for the Commit record will be seen in the in the record.

Example

Payment is made against Contract CON-009 and this record has 5 line items which exist in the SOV, but the payment has to be made only for Line item 1. The input XML will have all the data for this line item. After a successful integration, PayApp-010 gets created. When you open this record, all the 5 line items will be seen in PayApp-010.

Validation

Since the system adds all the line items from the SOV, which do not exist in the Input XML, the system performs additional checks at the time of creating the record.

- All auto-population and default values set for String fields will be honored at the time of creating the records. If no auto-population and default values exist for these required string fields then, the creation of record through integration will fail.
- All auto-population and default values for Numeric fields will also be honored. If no values exist then the system will default these fields to 0.
- Errors messages will be pertaining to the required field check validations.

Create Complete BP Record

Description

This method can be used to create new business process records with attachments in Unifier based on Integration design that has been defined in uDesigner. It allows attachments to be added to the upper form and line items.

> **Note**: If you want to create a BP record with line item information with attachments, then you should use this service. If this is a cost related BP, line items will be rolled up to the cost sheet.

Support

This process supports all type of BPs

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject createCompleteBPRecord(String shortname, String authcode, string projectNumber, String BPName, String BPXML, String iszipfile, FileObject[] fileobjects);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier. If you want to create a BP record at Company level then pass null for this parameter.
BPName	Identifier of the business process in Unifier, for example: Invoice. This is case sensitive, and should be the name as given in uDesigner
BPXML	identifier for BP content in XML format

Parameter	Description
fileobjects	Array of FileObjects, each FileObject is made up of two components: String filename (this should be the same name as in the record data <_attachment><filename> tag, and javax.activation.DataHandler as filestream. Files can be sent as a single zipped file or individual files.
isZipFile	Identifies if the file provided under the fileobjects tag is a zip file. The value for this tag is "yes" or "no". Yes to allow zip file as an attachment. Otherwise, select No. **Note:** Only .zip file format is acceptable. When using .zip file as an attachment, you should attach one zip file at a time that includes a single attachment of each file in the zip file.

BPXML Elements

The elements that include as part of BPXML parameters is dependent on the Integration interface design of your BP in uDesigner. XML format can be exported (downloaded) from uDesigner. Elements with direction marked as "Input" or "Both" will be part of XML format.

Note that elements lists should be send based on the type of BP and your design in uDesigner.

Following a list of elements that can be included as part of integration design in uDesigner and behavior when you include them. Note that elements lists should be sent based on the type of BP for which you are creating a record.

Tag Name	BP Type	Description
<amount>	Cost	Amount is a required field if you are sending line item information under _bp_lineitems. You will see same value on Unifier.
<status>	Any	a) If BP uses a status field then you can send status tag. b) If the BP is a workflow based then the record will be created and will stay in the initial status that is valid. User has to go through remaining workflow process. c) If it is non-workflow based then the message can send a valid status and the record will take that status. If there is only one status defined for the BP then by default that status will be picked even if user sends a wrong status value.

Tag Name	BP Type	Description
<record_no>	Any	You can send record number by using this tag. If you do not specify a record_no tag as part of message then a new record will be created. Unifier will assign a record number. If you specify a record_no then system will create a record with the given record number. If the record number is a duplicate number then an error message will be send back as response. If the record_no is specified and if it does not exists then a new record will be created.
attachment	Simple BP, Text BP, Line Item BP, Cost BP, Document Type BPs	Multiple attachment elements can be added to the upper form and to line items (if applicable). The attachment element contains sub elements that identify: File name Title Revision Number Issue Date Version Reference
<_comment>	Text BP	Enter the responses (comments) associated with the BP record.

Tag Name	BP Type	Description
<li_group_name>	All BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.	Use to specify group name.

Return Value

See Return Values. This is dependent on which WSDL you are going to use. If you use WSDL –1 then you should expect XMLObject as return value and if you use WSDL – 2 then you should expect XMLFileObject as return value.

Create Complete BP Record (Additional Information)

This topic includes additional information about creating complete BP records.

Acceptable File Formats

Only .zip file format is acceptable. When using a Zip file as an attachment, you should attach one zip file at a time that includes the single attachment for each file in the zip file. To get the BP XML structure, first send a message to get BP record. Refer to the Web Service getBPRecord for more information.

This interface can be extended with additional data elements. In order for Unifier to recognize those elements, the data element name must match those created in uDesigner for this business process.

YTB/AFC

If the tags YTB or AFC are present in the line items, Unifier processes these tags. If they are not present, Unifier will check the uDesigner BP setup configuration for Cost Adjustment. If this field has been set, the line item amount is treated as YTB/AFC and processed.

Single Record BP

Record Exists

If you send a message for single record BP and a record already exists then Unifier will not send back and error but if you send an invalid record number then Unifier will send back an error.

Record does not exist

If you send a message for single record BP and a record does not exist then Unifier will create a record for you. If the BP type is of Line item then any line items included in the message will be added to the BP record. If the BP type is of Simple then any line items added as part of message will be ignored.

If your BP has **uuu_creation_date** then it will be populated when you send a message.

If your Cost BP has fund related information then you have to send Fund code details. If you are using multi segment fund code then send the code segments with delimiters.

Example

Fundseg1-Fundseg2-Fundseg3.

The fund information send as part of Cost BP will be rolled up to Funding Sheet or Cost Sheet.

All data element values provided as part of integration message will be validated against the action form selected as part of design in uDesigner. Validation includes required field, form validation etc..,

If the action form used for validation is auto-populating values from a Business Process picker then system will auto-populate these values if a valid value is provided for that Business Process Picker.

Attachments can be added to upper form and line items, and multiple records.

Files can be sent as a single zipped file or individual files for the complete transaction record, the same file can be referenced multiple times in the same transaction; however, files contained in the zip file should still be described by separate attachment elements.

API

This API requires Integration Design to be published in uDesigner for the business process.

Attachment file cross references is not supported. The reference sub element under attachments should always have the value ‘No’.

Adding attachments is optional. If attachment elements are not provided, then the API behaves exactly like createBPRecord API.

> **Note**: This API is not compatible with non Java web services client platforms.

Total attachment size per request across all files is limited to 250 MB. It is recommended that files are compressed into 1 zip file for submission to the API.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

The following XML tag <_refnum> </refnum> should be added under BP line item tag <_bp_lineitem> </bp_lineitem> to support the ability to import the line items for General Spends and Payment Applications.

Creating Complete BP Record for Document type BP with Folder Structure

The Web Services calls support creating new records and allows users to include the line items assigned to a particular folder.

- The tag <folder_path> specifies the path of the folder for the line item. This tag already exists in the current input XML for Document Type BP with folder structure.
- The folder path (<folder_path>) contains the names of the folders.
  - If the folders mentioned in the folder path do not exist, then the folders are created and the line items are added to the last folder mentioned in the folder path.
  - If the folder names already exist, then the line items are created under the last folder mentioned in the path.
- The Input integration XML contains the tags, as part of bp_lineitems elements.
- The syntax of the entry for the <folder_path> field is: "/<Folder Name1>/"
- If you create the line items under the root node, then the valid inputs within the tag are either "/" or blank.
- If the folder path is blank, then the line item is created at the root node "Attachments" and the folder name will be "Attachments".
- If the line items have attachments in the <_attachment> elements, then the line items are added to the respective folder.

When the folders are being created through Web Services, the validations (in web interface of Unifier) are completed during the naming of the folders. You can enter non-ASCII characters as folder names. For example, you can enter Chinese characters as folder names in the folder path.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Add BP Line Item

Description

This method adds a BP line item to an existing business process record based on Integration design that has been defined in uDesigner. For more information refer to the *Unifier uDesigner User Guide*.

This method can also be used to create new BP records with line items, check LineXML elements sub-topic to find out more.

Support

This process supports all type of BPs.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project or Shell Level	Yes

Prototype

public XMLObject addBPLineItem(String shortname, String authcode, string projectNumber, string BPName, String ItemXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or shell in Unifier
BPName	identifier of the business process in Unifier, for example: Invoice. This is case sensitive, and should be the name as given in uDesigner
ItemXML	identifier for line item content in XML format

ItemXML Elements

The elements that include as part of LineXML parameter are dependent on the Integration interface design of your BP in uDesigner. ItemXML format can be exported (downloaded) from uDesigner. Elements with direction marked as "Input" or "Both" will be part of XML format.

The following is a list of elements that can be added as part of integration design and their behavior when you include them. Note that elements lists should be sent based on the type of BP for which you are creating a record.

Tag Name	BP Type	Description
<amount>	Cost	Amount is a required field if you are sending line item information under _bp_lineitems. You will see same value on Unifier.
<CBS_code>	Cost	CBS Code is a required field if you are sending line information under _bp_lineitems.

Tag Name	BP Type	Description
<record_no>	Any	You can send record number by using this tag. If you do not specify a record_no tag as part of message then a new record will be created with the line item information that is provided as part of ItemXML If you specify a record_no then system will add new line item(s) based on the data provided as part of ItemXML If the record_no is specified and if it does not exists then a new record will be created with the line item information that is provided as part of ItemXML.
<status>	Any	You can send record status by using this tag. This tag is valid only for multi-record BPs. If the record_no is specified and if it does not exists then a new record will be created with the line item information that is provided as part of ItemXML. If the BP is a workflow based then the record will be created and will be created and will stay in the initial status that is valid. User has to go through remaining workflow process. If it is a non-workflow based then the message can send a valid status and the record will take that status. If there is only one status defined for the BP then by default that status will be picked even if user sends a wrong status value.
<li_group_name>	All BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.	Use to specify group name.

Return Value

See Return Values

Add BP Line Item (Additional Information)

This topic includes additional information about adding BP line items.

Interface

This interface can be extended with additional data elements. In order for Unifier to recognize those elements, the data element name has to match those created in uDesigner for this business process.

YTB/AFC

If the tags YTB or AFC are present in the line items, Unifier processes these tags. If they are not present, Unifier will check the uDesigner BP setup configuration for Cost Adjustment. If this field has been set, the line item amount is treated as YTB/AFC and processed.

To add a line item to company level BP record then send null for projectNumber parameter. If you send a project number for a BP that is at Company level then you will get an error that BP name is not correct.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

The following XML tag <_refnum> </refnum> should be added under BP line item tag <_bp_lineitem> </bp_lineitem> to support the ability to import the line items for General Spends and Payment Applications.

This service can be used to add lines to Payment Applications and General Spends type of business process. When adding lines to Payment Applications type of business process, existing lines if any will be removed and new lines will be added that are provided as part of the service call. For General spends line will be appended like any other business process.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Add Complete BP Line Item

Description

This method adds a BP line item to an existing business process record based on Integration design that has been defined in uDesigner. For more information refer to *Unifier uDesigner User Guide*. It allows multiple files to be attached to the BP line item.

This method can also be used to create new BP records with line items, check LineXML elements sub-topic to find out more.

Support

This process supports all type of BPs.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject addCompleteBPLineItem(String shortname, String authcode, string projectNumber, string BPName, String ITemXML , FileObject[] fileobjects);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or shell in Unifier
BPName	identifier of the business process in Unifier, for example: Invoice. This is case sensitive, and should be the name as given in uDesigner
ItemXML	identifier for line item content in XML format
fileobjects	Array of FileObjects, each FileObject is made up of two components: String filename (this should be the same name as in the record data <_attachment><filename> tag, and javax.activation.DataHandler as filestream. Files can be sent as a single zipped file or individual files

ItemXML Elements

The elements that include as part of LineXML parameter are dependent on the Integration interface design of your BP in uDesigner. ItemXML format can be exported (downloaded) from uDesigner. Elements with direction marked as "Input" or "Both" will be part of XML format.

The following is a list of elements that can be added as part of integration design and their behavior when you include them. Note that elements lists should be send based on the type of BP for which you are creating a record.

Tag Name	BP Type	Description
<amount>	Cost	Amount is a required field if you are sending line item information under _bp_lineitems. You will see same value on Unifier.

Tag Name	BP Type	Description
<record_no>	Any	You can send record number by using this tag. If you do not specify a record_no tag as part of message then a new record will be created with the line item information that is provided as part of ItemXML If you specify a record_no then system will add new line item(s) based on the data provided as part of ItemXML If the record_no is specified and if it does not exists then a new record will be created with the line item information that is provided as part of ItemXML.
<status>	Any	You can send record status by using this tag. This tag is valid only for multi-record BPs. If the record_no is specified and if it does not exists then a new record will be created with the line item information that is provided as part of ItemXML. If the BP is a workflow based then the record will be created and will be created and will stay in the initial status that is valid. User has to go through remaining workflow process. If it is a non-workflow based then the message can send a valid status and the record will take that status. If there is only one status defined for the BP then by default that status will be picked even if user sends a wrong status value.
attachment	Any	Multiple attachment elements can be added to the upper form and to line items (if applicable). The attachment element contains sub elements that identify: ▸ File name ▸ Title ▸ Revision Number ▸ Issue Date ▸ Version ▸ Reference

Tag Name	BP Type	Description
<li_group_name>	All BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.	Use to specify group name.

Return Value

See Return Values

Add Complete BP Line Item (Additional Information)

This topic includes additional information about adding complete BP line items.

Interface

This interface can be extended with additional data elements. In order for Unifier to recognize those elements, the data element name has to match those created in uDesigner for this business process.

YTB/AFC

If the tags YTB or AFC are present in the line items, Unifier processes these tags. If they are not present, Unifier will check the uDesigner BP setup configuration for Cost Adjustment. If this field has been set, the line item amount is treated as YTB/AFC and processed.

To add a line item to company level BP record then send null for projectNumber parameter. If you send a project number for a BP that is at Company level then you will get an error that BP name is not correct.

Attachments

Attachments can be added to line items.

Files can be sent as a single zipped file or individual files for the complete transaction record, the same file can be referenced multiple times in the same transaction. However, files contained in the zip file should still be described by separate attachment elements.

API

This API requires Integration Design to be published in uDesigner for the business process.

Attachment elements specified as part of upper form data will be ignored by this API.

Attachment file cross references is not supported. The reference sub element under attachments should always have the value ‘No’.

Adding attachments is optional. If attachment elements are not provided, then the API behaves exactly like addBPLineItem API.

> **Note:** This API is not compatible with non Java web services client platforms.

Total attachment size per request across all files is limited to 250 MB. It is recommended that files are compressed into 1 zip file for submission to the API.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

The following XML tag <_refnum> </refnum> should be added under BP line item tag <_bp_lineitem> </bp_lineitem> to support the ability to import the line items for General Spends and Payment Applications.

This service can be used to add lines to Payment Applications and General Spends type of business process. When adding lines to Payment Applications type of business process, existing lines if any will be removed and new lines will be added that are provided as part of the service call. For General spends line will be appended like any other business process.

Adding Complete BP Line Item for Document type BP with Folder Structure

You can add additional line items (including a document) into a particular folder in the Document Type BP with folder structure for an existing, or new, BP record.

- The tag <folder_path> specifies the path of the folder for the line item. This tag already exists in the current input XML for Document Type BP with folder structure.
- The folder path (<folder_path>) contains the names of the folders.
  - If the folders mentioned in the folder path do not exist, then the folders are created and the line items are added to the last folder mentioned in the folder path.
  - If the folder names already exist, then the line items are created under the last folder mentioned in the path.
- The Input integration XML contains the tags, as part of bp_lineitems elements.
- The syntax of the entry for the <folder_path> field is: “/<Folder Name1>/”
- If you create the line items under the root node, then the valid inputs within the tag are either “/” or blank.
- If the folder path is blank, then the line item is created at the root node “Attachments” and the folder name will be “Attachments”.
- If the line items have attachments in the <_attachment> elements, then the line items are added to the respective folder.

When the folders are being created through Web Services, the validations (in web interface of Unifier) are completed during the naming of the folders. You can enter non-ASCII characters as folder names. For example, you can enter Chinese characters as folder names in the folder path.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit,

and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Note: You can use this method to add line items to a BP record that does not have any line items. If a record already has the SOV line items that existed at the time of the record creation, then new line items cannot be added. If the call is started, then the system generates an error message.

Update BP Record

Description

This method updates a record in Unifier.

Support

This service is available at both Company Level and Project or Shell Level BPs of type non-workflow Line Item type and can only update information on upper form. Line items will be ignored. Additionally the service is also available at Project or Shell Level for Cost (Non-Payment type), Simple, Text type and Document Type (Basic and Advanced) business process. In these cases, too, the service can only update information on upper form. Line items will be ignored.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject updateBPRecord(String shortname, String authcode, String projectNumber, String BPName, String BPXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.

BPName	Identifier of the BP
BPXML	Content of the BP that needs to be updated

BPXML Elements

The BPXML Elements are based on BP design. So in order to use this method user has to first use getBPRecord service. getBPRecord service will provide user with valid XML elements that are part of BP design.

If you want to update record for a BP at company level then do not pass projectnumber parameter.

If you send invalid projectnumber then you will get an error.

<record_no> tag as part of <_bp> element is required while send updateBPRecord message.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Return Value

See Return Values

Sample Method

updateBPRecord("acme", "acme_authcode", "Proj-0001", "Blanket PO", "<XML Content>")

Update BP Record (Additional Information)

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Update Complete BP Record

Description

This method updates a record in Unifier. It also provides the ability to add attachments to the upper form. It will not delete existing attachments.

Support

This service is available at both Company Level and Project or Shell Level BPs of type non-workflow Line Item type and can only update information on upper form. Line items will be ignored. Additionally the service is also available at Project or Shell Level for Cost (Non-Payment type), Simple, Text type and Document Type (Basic and Advanced) business process. In these cases, too, the service can only update information on upper form. Line items will be ignored.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject updateCompleteBPRecord(String shortname, String authcode, string projectNumber, String BPName, String BPXML, String iszipfile, FileObject[] fileobjects);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
BPName	Identifier of the BP
BPXML	Content of the BP that needs to be updated
iszipfile	This tag identifies if the file provided under the fileobjects tag is a zip file or not. Values for this tag are "yes" or "no".
fileobjects	Array of FileObjects, each FileObject is made up of two components: String filename (this should be the same name as in the record data <_attachment><filename> tag, and javax.activation.DataHandler as filestream. Files can be sent as a single zipped file or individual files..

BPXML Elements

The BPXML Elements are based on BP design. So in order to use this method user has to first use getBPRecord service. getBPRecord service will provide user with valid XML elements that are part of BP design.

If you want to update record for a BP at company level then do not pass projectnumber parameter.

If you send invalid projectnumber then you will get an error.

<record_no> tag as part of <_bp> element is required while send updateBPRecord message.

Attachments can be added only to the upper form.

Files can be sent as a single zipped file or individual files for the complete transaction record, the same file can be referenced multiple times in the same transaction. However, files contained in the zip file should still be described by separate attachment elements.

API

This API requires Integration Design to be published in uDesigner for the business process.

Attachment file cross references is not supported. The reference sub element under attachments should always have the value 'No'.

Adding attachments is optional. If attachment elements are not provided, then the API behaves exactly like updateBPRecord API.

> **Note:** This API is not compatible with non Java web services client platforms.

Total attachment size per request across all files is limited to 250 MB. It is recommended that files are compressed into 1 zip file for submission to the API.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Return Value

See Return Values

Update Complete BP Record (Additional Information)

Updating Complete BP Record for Document type BP with Folder Structure

The existing Web Services call updates the Upper form elements and you can add attachments to the Upper form.

> **Note**: The Document type BP with folder structure does not have record-level attachments.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Update BP Record V2

Description

This option will allow external systems to update a business process record and advance it to next step.

Support

This service is available at both Company Level and Project or Shell Level BPs of type non-workflow Line Item type and can only update information on upper form. Additionally, the service is also available at Project or Shell Level for Cost (Non-Payment type), Simple, Text type and Document Type (Basic and Advanced) business process. In these cases, too, the service can only update information on upper form.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes
Program Level	No

Prototype

public XMLObject updateBPRecordV2(String shortname, String authcode, String shellNumber, String BPName, String BPXML, String options).

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
shellNumber	Identifier of the project or shell in Unifier.
BPName	Identifier of the BP
BPXML	Content of the BP that needs to be updated and moved to next step.

Parameter	Description
Options	This tag will allow user to send additional information related to workflow and line items. Following are the tags under options tag and their description ▸ LineItemIdentifier: Name of the DE on the detail form / line item to uniquely identify line items. ▸ WFCurrentStepName: Name of the current workflow setup ▸ WFActionName: Action that should be taken to move the WF to next step ▸ SYS Auto Sequence type DE

Return Value

See Return Values

Sample Method

updateBPRecordV2("acme", "acme_authcode", "PRJ-0001", "Contracts", <XML content>,<options>)

Update BP Record V2 (Additional Information)

This topic includes additional information about updating BP record V2.

Simple Object Access Protocol (SOAP)

To send SOAP services as XML, ensure that you follow the general rules of XML:

- All tag data (not CDATA) needs to be escaped for < > & "
- Escaping must be a part of the client code which generates the web service call

> **Note**: Those using SOAPUI to make a call must do it manually.

Unifier supports MTOM base64 encoded files in our Unifier Web Service SOAP platform for:

- createCompleteBPRecord
- updateCompleteBPRecord
- addCompleteBPRecord
- getCompleteBPRecord

Shell and Company Levels

If the Shell number is null then the record will be assumed for company level record.

The step involved in sending a record (Send To) is pre-assigned.

Options

- WFCurrentStepName

Value of this tag identifies the current step of the record. This is to ensure that web service call is called for correct step with correct action.

- WFActionName

  Value of this tag determines the action that should be taken. This action name will be validated against the WFCurrentStepName

WFActionName

If the option WFActionName tag is empty, then the system updates the record, only. WFCurrentStepName in this case is not validated.

If the option WFActionName tag has a value, then the system pushes workflow to the next step, based on the value of WFActionName tag. All form level validations will be triggered before pushing workflow to the next step. In this scenario WFCurrentStepName will be validated to make sure that the step is valid and action is valid. The option WFActionName tag will be ignored for non-workflow business processes.

Validations while processing WFCurrentStepName and WFActionName tags

- If WFActionName tag *is empty*, then do not validate WFCurrentStepName
- If WFActionName tag *is not empty*, then validate WFCurrentStepName

WFCurrentStepName

The record is still on the step identified by WFCurrentStepName. If it is not, the system generates an error.

Step identified by WFCurrentStepName value has "Enable Step for Integration" option set to "Yes". If it is not, the system generates an error.

Validate WFActionName is valid action that can be taken from step identified by the value of WFCurrentStepName.

After going through all validations rules, the system moves the record to next step. The Creator, who has been selected on the Admin > BP Setup > Workflow (node) > Auto Creation (tab) > Creator, is the person who takes action on the step.

It is possible that someone would have accepted the task on step identified by the value of WFCurrentStepName. (This can happen because "Assignees" option for this step under BP Setup is not disabled). There are three possible scenarios under this condition:

- No one accepted the task: In this scenario system will automatically accepts the task, update record and line items and move record to next step. System will remove task from all other users list.
- Someone accepted the task: In this scenario system will update record and move it to next step and will remove task from other users list including the use who accepted the task.
- Someone accepted the task and move workflow to next step. System will generate an error message as the task is already accepted and WF is moved out of that step

LineItemIdentifier

There are two scenarios for this option:

a) LineItemIdentifier value is empty: In this scenario system will add all line items. <_delete_bp_lineitems> tag will be ignored.

b) LineItemIdentifier value is not empty: In this scenario value provided as part of LineItemIdentifier should be a tag under _bp_lineitems. If not system will error out.

If LineItemIdentifier value is an element under _bp_lineitems, then the system will try to query up a line item based on the value provided under this element.

If the line is found then it will be updated.

If two lines are found, then the system generates an error (duplicate rows).

If the line is not found, then it will be added.

Notes:

- The mentioned behavior mentioned above is applicable to all lines on all tabs.
- The system will not update/remove/add line items of Cost type of Business Process records that have reached terminal status (not just end step), using this new service.

The "LineItemIdentifier" option is ignored for following type of business processes:

- Simple: Document with Detail form
- Text

WFActionName and **WFCurrentStepName**

This option is ignored for non-workflow Business Processes.

Import the line items for General Spends and Payment Applications

The following XML tag <_refnum> </refnum> should be added under BP line item tag <_bp_lineitem> </bp_lineitem> to support the ability to import the line items for General Spends and Payment Applications.

This service can be used to add lines to Payment Applications and General Spends type of business process. When adding lines to Payment Applications type of business process, existing lines if any will be removed and new lines will be added that are provided as part of the service call. For General spends line will be appended like any other business process.

Note: *updateBPRecordv2* cannot be used to update existing line items for Payment Applications type of business process.

Updating BP Record V2 for Document type BP with Folder Structure

The Web Service call supports the adding of line items to a specific folder to an existing Document type BP record.

The **Update BP Record V2** is applicable to Base Commit and Change Commit of *Summary Payment Application* type business processes.

The **Update BP Record V2** is partially supported for Payment Application business processes of Summary Payment Application type. Use this service to add standard tab line items to a payment record only when the record does not have any standard tab line items created, yet (in such case, the same service can also be used to optionally progress the workflow). Alternately, this service can be used only to progress the workflow on a record.

This service is not available to modify or delete standard tab line items.

Payment BP (both the classic and summary payment) line items belonging to tabs other than the standard tab can be added/modified/deleted by using: updatebprecordv2 call.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit and Change Commit of Summary Payment Application SOV Type BPs.

For more details about XML tags, see Create and Add Methods in the ***Create BP Record (Additional Information)*** (on page 96) section.

Update Methods (Update BP Record V2)

The Cost Allocation using the Update BP Record V2 method is necessary when Summary Line Item is added at the creation step and Actual Cost Allocation is added in the subsequent step.

When you need to add Cost Allocation to an existing Line Item, the existing “LineItemIdentifier” tag can hold the information of the Line Item that needs to be updated with the Cost Allocation

The LineItemIdentifier is the key tag for identifying the Line Item that needs to be updated.

You can add new Summary Line Items and Costed Line Items.

If the XML template has the Summary Line Item fields, along with the Cost Allocation, then the Summary Line Items need to be updated.

> **Note**: In this case, the LineItemIdentifier tag is the key.

For modifying Line Items, either by adding new Costed Line Items or updating existing Costed Line Items for Base Commit and Change Commit Business Process records, the following rules are applied:

- CBS Code forms the identifier for identifying the Costed Line Items that need to be updated in Base Commit BPs.
- For a Base Commit BP record:
  - If a CBS Code that is specified in the _cost_allocation block does not exist in the Line Item, then this will be treated as a new Costed Line Item for the specified Line Item.
  - If the CBS Code specified in the _cost_allocation block already exists, then this will be treated as an update and the Costed Line Item for the specified Line Item will be updated.
  - If there are multiple Cost Allocation blocks with the same CBS Codes, then update will not take place and results in a duplicate CBS Code error.

> **Note**: The above rule applies to updating the existing Line Items in the Change Commit BP, also.

New modifications to Line Items existing in the SOV can be done by specifying the sovlinum. For modifying the Costed Line Items in the Summary Line Items, the combination of sovlinum and CBS Code will be used.

Since the Short Description already exists for the Line Items, this is as an Optional field.

The Costed Line Items that need to be updated will be identified by the Cost Code/ bItemId.

All operations that are currently possible with the updateBPRecordV2 call will be applicable here, aloso.

Example

Workflow progress, deletion of line items etc.

Validation

If the input XML used for updating Line Items has only Cost Allocation details, along with the Line Item reference, then the Line Items update with the Cost Allocation will go through without any errors.

All validations for the required fields in the Summary level and the Cost Allocation level will be applicable.

All system level validations existing for the above mentioned BPs will be applicable.

Example

Costed amount = Line item amount at a given step

Existing validations for update BP record V2 will be applicable.

Data Elements in Line Items

If the following Data Elements are present in the Line Items, they will not be updated during UpdateRecordV2 call.

- `bitemid`
- `budgetid`
- `sovitemid`
- `uuu_cm0_picker`
- `xid`
- `ds_code`
- `fsm_lineitem_id`
- `asm_lineitem_id`
- `uuu_lat`
- `uuu_long`
- `uuu_company_account_name`
- `uuu_company_account_code`
- `uuu_company_acc_codepicker`
- `uuu_activity_picker`

- uuu_asset_picker
- ref_bpo_lineitem
- ref_bpo
- ref_rfb
- refid
- uuu_sovlinum
- otherCompanyID
- uuu_costcode_picker
- scheduled_value
- uuu_asset_cum_depreciation
- uuu_asset_curr_period_dep
- uuu_asset_net_book_value
- uuu_unit_price
- currencyid
- due_date
- uuu_asset_calc_as_of_date
- uuu_from_date
- uuu_issue_date
- uuu_last_update_date
- uuu_rfb_due_date
- uuu_to_date
- uuu_week_picker
- uuu_datasource
- uuu_depreciation_mtd
- uuu_line_item_status
- row_id
- uuu_bid_count
- uuu_bidders_count
- uuu_planning_item_picker
- uuu_proj_picker
- record_no
- uuu_resc_picker
- uuu_role_picker
- uuu_commit_short_desc
- end_date
- uuu_creation_date
- uuu_asset_name
- uuu_commit_breakdown
- uuu_asset_code
- uuu_asset_navigation_code
- uuu_timescale_units

- creator_id
- wpid
- uuu_cost_costattribute
- uuu_cost_external_refid
- uuu_cost_status
- uuu_cost_cost_type
- uuu_cost_description
- uuu_cost_item
- uuu_cost_owner
- uuu_cost_code
- uuu_dm_create_date
- uuu_file_create_date
- uuu_file_issue_date
- uuu_dm_percent_complete
- uuu_file_size
- uuu_file_version
- uuu_dm_description
- uuu_dm_node_path
- uuu_dm_node_name
- uuu_file_revision_no
- uuu_file_title
- uuu_dm_create_by
- uuu_file_create_by
- uuu_fund_fundcategory
- uuu_fund_long_desc
- uuu_fund_description
- uuu_fund_fundname
- uuu_fund_code
- uuu_description
- uuu_resc_nwd_type
- uuu_resc_interest
- uuu_resc_proficiency
- uuu_resc_skill
- uuu_role_status
- uuu_role_name
- uuu_resc_capacity
- uuu_user_address
- uuu_user_city
- uuu_user_country
- uuu_user_email
- uuu_user_fax

- uuu_user_firstname
- uuu_user_homephone
- uuu_user_lastname
- uuu_user_mobilephone
- uuu_user_pager
- uuu_user_state
- uuu_user_timezone
- uuu_user_title
- uuu_user_workphone
- uuu_user_zip
- uuu_resc_status
- uuu_resc_code
- uuu_resc_name
- uuu_early_finish
- uuu_early_start,uuu_finish
- uuu_late_finish
- uuu_late_start
- uuu_start
- uuu_act_pct_complete
- uuu_activity_work_hrs
- uuu_duration
- uuu_fixed_cost
- uuu_float
- uuu_labor_cost
- uuu_non_labor_cost
- uuu_total_cost
- uuu_activity_id
- uuu_flag_complete
- uuu_flag_milestone
- uuu_activity_resources
- uuu_dependency_type
- uuu_activity_name
- uuu_outline_code
- uuu_cost_li_type

Get BP Record

Description

This method gets a specific Business Process record in Unifier based on Integration design that has been defined in uDesigner. For more information refer to *Unifier uDesigner User Guide*. This method works for both company level and project or shell level BPs.

Support

This process supports all type of BPs.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject getBPRecord(String shortname, String authcode, string projectNumber, String BPName, String recordNumber);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier. Pass null for Company Level BP
BPName	identifier of the business process in Unifier, for example: Invoice
recordNumber	identifier of the business process example: inv-0001

Return Value

See Return Values

For Text BPs, the contents of the <_comment> tag, which contains the response list (comments), is also returned.

Sample Method

getBPRecord("acme", "acme_authcode", "proj-01", "Invoice","innv-0001") will return the XML Record of the BP.

Get BP Record (Additional Information)

This topic includes additional information about getting a BP record.

Use the <li_group_name> tag to specify group names. This tag is applicable to all BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.

recordNumber

If the recordNumber is left blank or invalid recordNumber then the method will returns the BP's skeletal XML. BP template can also be downloaded from uDesigner.

To get company level BP record information, send null for projectNumber parameter. If you send a project number for a BP that is at Company level then you will get an error that BP name is not correct.

To get Single Record BP you need a record number populated on the record. If you do not have a record number populated then getBPRecord will return back the structure of the BP.

If you getting information for a Cost Type BP then you will always get <bItemID> and <fund_code> elements as part of return XML.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

When this method is used for the above mentioned BPs, the response file will contain Cost Allocation details for various Line Items.

Get BP List

Description

This method gets all the records for a specific Business Process in Unifier based on Integration design that has been defined in uDesigner. For more information refer to the "uDesigner" section in the *Unifier Administration Guide*.

Support

This process supports all type of BPs.

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject getBPList (String shortname, String authcode, string projectNumber, String BPName, String[] fieldnames, String filterCondition, String[] filterValues);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or shell in Unifier
BPName	identifier of the business process in Unifier, for example: Invoice
fieldnames	names of the elements in the BP that you want to retrieve; this is an optional parameter. If not specified, all elements will be returned. Currently this parameter is not supported through Web services.
filterCondition	a list of supported condition, this is an optional parameter. If not specified, no conditions will be applied. Field names that you enter as filterCondition should match with the elements that you have selected while designing Integration Interface in uDesigner.
filterValues	a list of values depending on the filter conditions. Currently this parameter is not supported through Web services.

fieldnames

Values that you are going to send through fieldnames parameter are dependent on your BP design. Only those filed can send through fieldnames that are part of your BP design. You can send a name from line item list.

Return Value

See Return Values

Sample Method

getBPList("acme", "acme_authcode", "proj-01", "Invoice",null,null,null) will return the XML List of all the Records of the BP Invoice.

```
<gen:getBPList>
       <!--Optional:-->
         <gen:shortname>4C</gen:shortname>
         <!--Optional:-->
         <gen:authcode>123</gen:authcode>
         <!--Optional:-->
         <gen:projectNumber></gen:projectNumber>
         <!--Optional:-->
         <gen:BPName>Vendors</gen:BPName>
         <!--Zero or more repetitions:-->
```

```
<gen:fieldnames></gen:fieldnames>
<!--Optional:-->
<gen:filterCondition></gen:filterCondition>
<!--Zero or more repetitions:-->
<gen:filtervalues>showlineitems=yes</gen:filtervalues>
```

Or: `<gen:filtervalues>showlineitems=`**no**`</gen:filtervalues>` for Upper form details, only.

```
</gen:getBPList>
```

Get BP List (Additional Information)

This topic includes additional information about getting a BP list.

Company-level BP Record

To get company level BP record information, do not send projectNumber parameter. If you send project number and the BP exists at company level then you will get an error that BP name is not correct.

Single Record BP

To get Single Record BP you need a record number populated on to the record. If you do not have a record number populated then getBPRecord will return back the structure of the BP.

filterCondition

If you cannot send filterCondition and filter values currently then it is recommended that these be set to "null." If you want to filter out the records that you want to retrieve then you can use this element. If you want to include more than one filter condition then following this syntax

<_filterCondition>fieldname=value && fieldname=value<_filterCondition>

You can add custom field clause through filters.

Example

end_date=04-13-2005, && record_no=uas-1200

If condition value is string (ex: ABC Inc) then value should be entered as follows <_filterCondition>Vendor Name='ABC Inc'</_filterCondition>. Observe that the value has single quotes. Only equal to "=" operation is supported right now.

> **Note:** If errors are present when evaluating the basic filter condition system will ignore it, but if there are errors in custom filter condition field names and values, system will return an empty list. All condition fields are evaluated using AND. If the combination has only basic fields and it has an error in processing all the records will be returned.

filterNames

The Filter name will contain list of elements that you want to see as part of output XML. Ex: If you want to see project name and project number as output then include projectname and projectnumber.

filterValues, and filterNames are *not supported* as part of Web services for this release.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

li_group_name

Use the <li_group_name> tag to specify group names. This tag is applicable to all BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

When this method is used for the above mention BPs, the response file will contain Cost Allocation details for various Line Items.

Get Complete BP Record

Description

Methods returns back complete BP record information. This includes any attachments, general comments etc.

Support

This service is available at both Company Level and Project and Shell Level BPs

Installation

ASP and Self host

Level	Yes or No
Company Level	Yes
Project and Shell Level	Yes

Prototype

public XMLObject getCompleteBPRecord(String shortname, String authcode, String projectNumber, String BPName, String record_no);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string
projectNumber	Identifier of the project or shell in Unifier.
BPName	Identifier of the BP
record_no	A valid record number.

Return Value

See Return Values

Sample Method

getCompleteBPRecord("acme", "acme_authcode", "Proj-0001", "Invoices", "Inv-0001")

Get Complete BP Record (Additional Information)

This topic includes additional information about getting a complete BP record.

Use the <li_group_name> tag to specify group names. This tag is applicable to all BPs that support Summary line items other than Summary Payment Application BPs of Base Commit, Change Commit and Payment Applications.

Company-Level BP Record

If you want to get company level BP record information then pass null for projectnumber element.

You will get a zip file as an output. The zip file contains any attachments and a PDF file that contains BP record information.

<record_no> tag as part of <_bp> element is required while send getCompleteBPRecord message.

This service can be used for both Project (Standard) and Shells of cost code type CBS and Generic.

Getting Complete BP Record for Document type BP with Folder Structure

The existing Get Complete BP record displays the <folder_path> in <bp_lineitems> elements and <attachments_folder> elements. The folder path seen in the line items elements matches the folder path seen in the <attachments_folder>. The <folder_name> is displayed with the <folder_path> in the <attachments_folder> section.

Behavior Specific to Summary Payment Application SOV Type BPs

> **Note**: The information is applicable to Base Commit, Change Commit, and Payment Application Business Processes of Summary Payment Application SOV Type BPs.

When this method is used for the above mentioned BPs, the response file will contain Cost Allocation details for various Line Items.

CBS Code Methods

This section covers the following:

- Get CBS Structure
- Create CBS Code
- Get CBS Codes
- Update CBS Codes

> **Note**: The Cost Breakdown Structure (CBS) methods have replaced the Work Breakdown Structure (WBS) methods; however, for legacy purposes the initials "WBS" have remained as is (for example, *WBS*XML in the Create CBS Code method is the identifier used for the CBS creation XML format).

In This Section

Get CBS Structure 135
Create CBS Code 136
Get CBS Codes 138
Update CBS Codes 139
Sort Cost Sheet 140

Get CBS Structure

Description

This method gets CBS Structure in Unifier for a company.

Support

This service is available at Company Level

Level	Yes or No
Company Level	Yes
Project and Shell Level	No

Prototype

Public XMLObject getWBSStructure(String shortname, String authcode);

Parameters

Parameter	Description
Shortname	identifier of the company, company's short name
Authcode	authentication key for the company, in text string

Return Value

See Return Values

Sample Method

getWBSStructure ("acme", "acme_authcode")

Create CBS Code

Description

This method creates CBS code under Project or Shell (cost code type as CBS) cost sheet.

Prototype

Public XMLObject createWBS(String shortname,String authcode, string projectNumber, String WBSXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or a shell in Unifier
WBSXML	identifier used for CBS creation XML format

WBSXML Elements

The following is a list of elements that are supported while creating CBS code. Any other codes that are defined as part of Cost Attribute form will be supported except User and BP Picker

Parameter	Description
<status>	Status is not a required tag when you are sending createWBS message. But if you include this tag then following are the valid values Active: Code will be created with Active status Inactive: Code will be created with Inactive status If you do not send this tag then code will be created without any status which is equivalent to Inactive status.
<cost_type>	This is not a required tag when you are sending createWBS message. But if you include this tag then following are the valid values Capital: To create CBS of type Capital Expense: To create CBS of type Expense If you do not send this tag then code will be created with Capital as Cost Type.
<costattribute>	This is not a required tag when you are sending createWBS message. But if you include this tag then send data that is defined as values under Cost Attribute data definition.

Return Value

See Return Values

Additional Information

You can create CBS Code with a Tree Structure.

getWBSStructure

Use getWBSStructure service to get the CBS attribute structure. The retrieved structure can be used to create new CBS codes. User can define a Cost Attribute form in uDesigner. In order to send XML message to create new Cost codes user has to send data using schema that is generated as output from getCBSStrucutre service.

If the Project or Shell CBS structure is in tree mode, or a parent / child relationship is required, the CBS code passed via the integration interface must concatenate the parent CBS code and child CBS code with "~~" as the delimiter.

If you have multilevel parent child relationship then you have to pass in "~~" as delimiter at all levels to separate parent and child.

Parent will be automatically created when you create child.

CBS Item element will take value of Description element.

This service will only work for Shells with Cost Code: CBS

Get CBS Codes

Description

This method gets the CBS Codes in Unifier for a particular project or shell.

Support

This service is available at Project/Shell

Level	Yes or No
Company Level	No
Project and Shell Level	Yes

Prototype

Public XMLObject getWBSCodes(String shortname, String authcode, String projectnumber, String options);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectnumber	Identifier of the project or shell in Unifier.
options	Three options are available: <status>: can be set to active or inactive. If status is blank or not provided it will return all status. <cost_type>: can be set to capital or expense. If cost_type is blank or not provided it will return all cost types. <summary_detail> : can be set to true or false. Setting to true, leaving blank or not providing the value will return all summaries and detail CBS codes. Setting to false will return on the detail CBS codes with no summaries.

Return Value

See Return Values

Sample Method

getWBSCodes ("acme", "acme_authcode", "P-00001" status=active)

Update CBS Codes

Description

This method updates CBS code under Project or Shell (cost code type as CBS) cost sheet. The updateWBS allows existing CBS attributes to be changed. All user-defined and system-attributes are available to be modified. The CBS Code cannot be edited.

Support

This service is available for Project/Shell.

Prototype

Public XMLObject updateWBS(String shortname,String authcode, string projectNumber, String WBSXML);

Parameters

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or a shell in Unifier
WBSXML	identifier used for CBS update XML format

Return Value

See Return Values

Additional Information

You can update CBS Code with a Tree Structure.

getWBSStructure

Users will be able to update CBS codes with tree structure.

Use getWBSStructure service to get the CBS attribute structure. The retrieved structure can be used to update CBS codes; however, the retrieved structure XML is needed to be wrapped as follows:

```
<![CDATA[
<List_Wrapper>  <!-- List Wrapper Added -->
<_bp_wbscode>
  <wbs_code></wbs_code>
  <wbs_item></wbs_item>
  <description></description>
  <costattribute></costattribute>
  <external_refid></external_refid>
  <owner></owner>
  <status></status>
  <cost_type></cost_type>
  <exposed_to_p6></exposed_to_p6>
</_bp_wbscode>
</List_Wrapper> <!-- List Wrapper Closed-->
]]>
```

User can define a Cost Attribute form in uDesigner. In order to send XML message to update Cost codes user has to send data using schema that is generated as output from getWBSStrucutre service.

If the Project or Shell CBS structure is in tree mode, or a parent / child relationship is required, the CBS code passed via the integration interface must concatenate the parent CBS code and child CBS code with "~~" as the delimiter.

If you have multilevel parent child relationship then you have to pass in "~~" as delimiter at all levels to separate parent and child.

This service will only work for Shells with Cost Code: CBS and standard projects.

Sort Cost Sheet

Description

This method sorts the project Cost Sheet on CBS code.

Support

This method only supports sorting of CBS code for Project/Shell.

Level	Yes or No
Company Level	No
Project Level and Shell Level	Yes

Prototype

Public XMLObject sortCostSheet (String shortname, String authcode, String projectNumber, String type, String sortOrder);

Parameters

The following is a list of parameters.

Parameter	Description
shortname	identifier of the company, company's short name
authcode	authentication key for the company, in text string
projectNumber	identifier of the project or a shell in Unifier

The following is a list of optional parameters.

Parameter	Description
type	Identifier for the column (the column that the sorting must happen). The only supported value is: CBS. This is an optional parameter, and if the parameter is not present the project Cost Sheet will be sorted based on CBS codes.
sortOrder	Identifier to define the sort order (ascending or descending). The values are: 'asc' or 'desc'. This is an optional input parameter. By default, the sort order is ascending.

Return Value

The Return values for this service are:

Code	Description
200	Ok
602	Project/Shell Number is not correct.
605	No Budget set up for the project.
500	Server Error, contact the system administrator.
1059	The type name does not match the column name from cost sheet.
1060	Invalid sort order. You must provide a value for either 'asc' or 'desc'.

Sample Method

sortCostSheet ("acme", "acme_authcode", "CBS","desc");

Additional Information

The following are the rules for sorting:

- For a flat structured Project Cost Sheet, there will be a simple alphanumeric sort on all the segment codes. The sorting will be case insensitive.
- For a tree structured Cost Sheet, the level sorting will happen first (the parent level first followed by the child level). Further alphanumeric sorting will happen within the level. The sorting will be case insensitive.

Note: The user must not perform any action on the project Cost Sheet when `sortCostSheet webservice` is running.

Business Process (BP) Record Data Processing Rules

Data Definition tags which have invalid value names will produce an error. If the tag is not sent, default value 0 is set if the type of the field is an integer.

Character fields: check for length.

Amount tag, where specified, checks for valid amount

For Cost type BP's, CBS code must be there.

For cost BP's, if the status tag is not given, Unifier scans to check if the BP has only one status value. If it has only one, Unifier will assume it as the default value for this record. If Unifier finds more than one status value, an error is reported.

Error checking is currently supported for Cost BP's only.

Data element of type Pull down will always take label value when a message is sent to Unifier.

For Business process that involved fund information, FBS code will be validated.

Any transaction against inactive CBS code will be rejected.

If a business process form has dynamic data set definition then data combination will be validated while creating or updating records through Integration.

Business Process (BP) Picker Support

Following Web Services calls will support Business Process Picker:

- `createBPRecord`
- `updateBPRecord`
- `addLineItem`
- `getBPRecord`
- `getCompleteBPRecord`
- `getBPList`

The following validation will be performed when a value is send for BP Picker data element:

- System will assume that the value send through Integration message is the value for Source Element selected as part of reference process definition.
- One and only one combination exists.
- If the combination is not valid or results into two records from reference business process, then integration will error out.

Similar process will be followed for the Line Item Picker on Detail Form.

General Validation Rules Across all Services

General commenting is only supported by `getCompleteBPRecord` service.

Special characters must be escaped in a valid XML document. The following characters `", ', < , >, &` will be escaped to:

- `&quot;`
- `&apos;`
- `<`
- `>`
- `&`

For Web Services (`createBPRecord`, `addBPLineItems`, `updateBPRecord`, `createProject`) the XML is scanned for validity and any errors are reported when found.

A non-required tag like <hello>hello</hello> is completely ignored.

Date fields (which are set as Date type) must be in the following format:

- `MM/dd/yyyy`
- `MM-dd-yyyy`
- `MM/dd/yyyy HH:mm:ss`
- `MM-dd-yyyy HH:mm:ss`
- `MM-dd-yyyy HH:mm:ss z (this format accepts timezone)`
- `MM/dd/yyyy HH:mm:ss z (this format accepts timezone)`
- `yyyy/MM/dd HH:mm:ss`
- `yyyy-MM-dd'T'HH:mm:ss`
- `yyyy-MM-dd`

`MM-dd-yyyy hh:mm` is a 24-hour clock. Users should specify time in hours instead of AM/PM notation.

The "z" in `MM/dd/yyyy HH:mm:ss z` signifies timezone.

Unifier converts user specified time to server time zone and stores in the database.

Tag names are *case sensitive*. For example: <_BP> is not a valid tag.

When designing Business Processes (BPs) that are related to Integration:

- All BPs must have the `record_no` tag in the Upper forms.
- For Cost related BPs, the Amount field must be present in the Line Items.

If the XML has a leading tag `<?xml encoding="iso-8859-1" ?>, Unifier` supports all characters within this ISO standard.

There are no form-level validations (required field, form level validation rules, and so forth).

There is no auto-populate feature through Integration, if integration interface is not designed for the BP.

General Limitations Across all Services

The `getBPList` service always returns the label data for a pull-down field.

The `getBPList` service does not return the values.

You cannot send fund related information for Cost Type BPs such as **General Spends** and **Payment Applications**.

Only the following ISO tags are supported as part of Web Services message:

–	NB SP 00 A0 160	¡ 00 A1 161	¢ 00 A2 162	£ 00 A3 163	¤ 00 A4 164	¥ 00 A5 165	¦ 00 A6 166	§ 00 A7 167	¨ 00 A8 168	© 00 A9 169	ª 00 AA 170	« 00 AB 171	¬ 00 AC 172	SH Y 00 AD 173	® 00 AE 174	¯ 00 AF 175
B–	° 00 B0 176	± 00 B1 177	² 00 B2 178	³ 00 B3 179	´ 00 B4 180	µ 00 B5 181	¶ 00 B6 182	· 00 B7 183	¸ 00 B8 184	¹ 00 B9 185	º 00 BA 186	» 00 BB 187	¼ 00 BC 188	½ 00 BD 189	¾ 00 BE 190	¿ 00 BF 191
C–	À 00 C0 192	Á 00 C1 193	Â 00 C2 194	Ã 00 C3 195	Ä 00 C4 196	Å 00 C5 197	Æ 00 C6 198	Ç 00 C7 199	È 00 C8 200	É 00 C9 201	Ê 00 CA 202	Ë 00 CB 203	Ì 00 CC 204	Í 00 CD 205	Î 00 CE 206	Ï 00 CF 207
D–	Ð 00 D0 208	Ñ 00 D1 209	Ò 00 D2 210	Ó 00 D3 211	Ô 00 D4 212	Õ 00 D5 213	Ö 00 D6 214	× 00 D7 215	Ø 00 D8 216	Ù 00 D9 217	Ú 00 DA 218	Û 00 DB 219	Ü 00 DC 220	Ý 00 DD 221	Þ 00 DE 222	ß 00 DF 223
E–	à 00 E0 224	á 00 E1 225	â 00 E2 226	ã 00 E3 227	ä 00 E4 228	å 00 E5 229	æ 00 E6 230	ç 00 E7 231	è 00 E8 232	é 00 E9 233	ê 00 EA 234	ë 00 EB 235	ì 00 EC 236	í 00 ED 237	î 00 EE 238	ï 00 EF 239
F–	ð 00F0 240	ñ 00F1 241	ò 00F2 242	ó 00F3 243	ô 00F4 244	õ 00F5 245	ö 00F6 246	÷ 00F7 247	ø 00F8 248	ù 00F9 249	ú 00FA 250	û 00FB 251	ü 00FC 252	ý 00FD 253	þ 00FE 254	ÿ 00FF 255
	—0	—1	—2	—3	—4	—5	—6	—7	—8	—9	—A	—B	—C	—D	—E	—F

In This Section

Pickers 150

Pickers

The following pickers are not supported through Web Services call:

- Commit Short Description
- Asset Picker
- Account Code Picker
- Planning Item Picker
- Resource Picker
- Project Picker
- Activity Picker
- Role Picker
- User Picker

Auto-populate from following attribute forms is not supported:

- Cost Attribute Form
- Fund Attribute Form
- DMS Attribute Form

For all Web Services that allow the addition of attachments, only zip format is supported only if the `iszipfile` tag is set to "yes". The `gzip` files (including the similar file formats) are not supported.

Company Picker

- Both the PUT and GET methods of the REST API support the Company Picker field.
- In the PUT calls, when the tag includes value for a Company Picker field, post validation values are inserted into the system. Similarly the GET calls will retrieve values for the Company Picker fields in the response output. This is applicable to both Upper Form and Detail Form data elements of the Company Picker type.

Known Issues Across all Services

The `createWBS` service is case insensitive. The service creates a CBS code with A-B-C-D and a-b-c-d.

Any transaction against inactive FBS code will not be rejected.

Auto-population of the Amount field on header form (from Line Items), is only supported if it is included in the form during the form design in uDesigner. Refer to the *Unifier uDesigner User Guide* and *Unifier Administration Guide* for more information.

The Shell name can be the same under different parents. In this scenario, if a user tries to create a business process record by passing in a value for a Shell picker data element, then the system displays an error (more than one Shell with same name).

Creating and Updating Schedule Sheet

A Schedule Sheet under Unifier Schedule Manager supports the following objects. For more information, refer to the *Unifier User Guide*, *Unifier Administration Guide*, and *Unifier uDesigner User Guide*.

- Activities
- Resource Assignments
- Activity Dependencies (Successors and Predecessors) along with relationship (Start to Finish, Finish to Start, etc.) and lag information.
- Association of an activity with a cost code (CBS Code) from Unifier Cost Manager.

Integration with P6 involves creation, modification and association of these objects automatically through Web Services calls.

As of the current Unifier release, integration with P6 Server supports the following information for each object.

- Activities: All elements including User Defined Fields.
- Resource Assignments: All elements including User Defined Fields and Unit of Measure
- Activity Dependencies: Based on Activity Dependencies option
- Association with Cost Codes: Unifier uses special logic as part of integration to build CBS Code to match with Unifier CBS Code or Cost Codes under Cost Manager. Refer to Associate CBS Codes with Activities section in this document.

To create and update these objects as part of Integration, the user must first create a data mapping relationship between data elements of Unifier and Data Elements of P6. The following section explains Data Mapping concepts and how to create one.

Apart from data mapping, Unifier should also know how to identify Activities and Resources that are coming from Primavera Project. This is to allow Unifier to support update web service method. Unifier currently supports two ways to identify an Activity and one way to identify a Resource.

An Activity can be identified by a GUID. Each activity in Primavera Project has a GUID. This GUID will be stored when an activity is created in Unifier and can be used later as part of update process.

The second way to identify an activity is through Activity ID. Each activity in Primavera Project has an Activity ID. This ID will be stored when an activity is created and will be used later as part of update process.

Resources are identified by their name. Each resource in Primavera Project has a name and this will be stored when a resource assignment is created in Unifier.

Data Mapping

To create and update these objects as part of Integration, the user must first create a data mapping relationship between Unifier and P6 data elements. The user can create one or more data mapping for each Schedule Sheet.

To define data mapping:

1) Navigate to your Project/Shell and open a schedule sheet.
2) From the menu bar, choose **File** > **Data Mapping**. The Data Mappings window opens.
   The Data Mappings window lists any mappings that you have created, and their default statuses. At run time, Unifier uses the mapping status and prevents the Select Default Mapping window from opening.
3) Click the **Add** button. The Data Mapping window opens.
4) On the **General** tab, enter the name and the description for the data mapping and click **Apply** to save changes and keep the window open, or click **OK** to save changes and close the window.
5) On the **Activity** tab, you can map Unifier schedule sheet columns to external CSV headers and XML tags for the import or export of activity information. Choose data elements from Activity and Resource Attribute forms. See the **Activity Tab Fields** table for details.
6) On the **Resources** tab, you can map resource information through XML integration. Enter the header in the XML element that corresponds to the column. Mapping resources is mandatory only if you select Import Resource Assignments on the Options tab.
7) On the **Options** tab you can configure XML options that the system uses when importing data. See the **Options Tab Fields** table for details.
8) Click **OK**.

One of the data mappings can be selected as default.This defaulting concept will be used while processing a web services call. How to use a data mapping and where default data mapping is used is explained in Web Service Methods section.

Data mapping can be marked to override existing data or merge existing data. Based on the web services call and business need appropriate option must be selected.

Activity Tab Fields

Data format	Data mapping requirements
Column	Select the data elements you want to import. Minimum required fields for XML import are Activity ID, Activity Name, Start Date, Finish Date, and Duration.
CSV Header	Enter the CSV header from the external source.

Data format	Data mapping requirements
XML Element	Enter the corresponding XML from the external source

Options Tab Fields

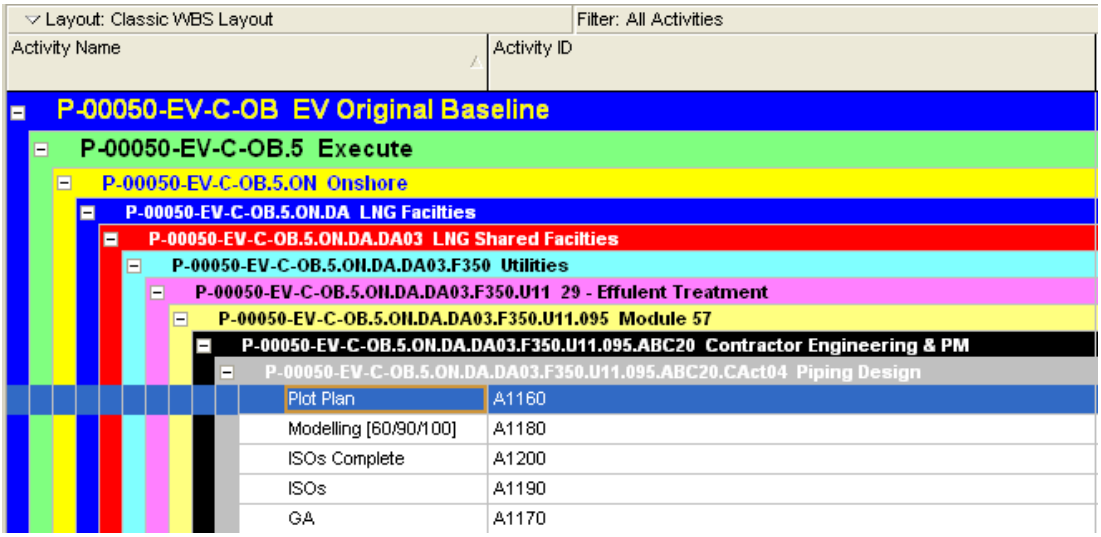
In this field:	Do this:
XML Import	Define options for importing XML activity schedules into Unifier. Mapping options are the same for Primavera XML and Microsoft Project XML.
Options	You can either retain existing schedule information in Unifier, or overwrite it complete upon importing an external XML file. 1. Merge into existing schedule. With this selection, you have a sub-option to Delete Activities removed from the source schedule. 2. Overwrite existing schedule replaces the existing schedule.
Data Elements	Select the appropriate checkboxes if you want to: - Import activities (rows) - Include dependencies - Assign resources - Assign CBS codes
Data Elements *(continued)*	1. Activity Dependencies: Select this checkbox to retain activities from the XML source file. 2. Activity Calendar: Select this checkbox to retain the activity calendar from the XML source file. If imported, the activity calendar will trigger the recalculation of activity dates as needed, and will override any existing activity calendar association. If a calendar is not imported, the activity will use the existing calendar defined in the Schedule Sheet properties.

In this field:	Do this:
Data Elements *(continued)*	There must be a calendar in Unifier with the same name as the calendar in the import file. (See "Creating Multiple Calendars".) 1. Resource Assignments: Select this checkbox to retain the resource assignments from the XML source file. If you want to import resources, define the resource types for the data definition SYS Resource Type. For Microsoft Project files, use standard resource types: Work, Material, and Cost. Upon import, these resource types will soft book. 2. CBS Codes: Select this checkbox to import CBS codes 3. Number of Levels: If you choose to import Primavera CBS codes, you can specify the segments that should be considered in the Primavera XML file for the codes (from 1 to 9) and the CBS code suffix mask. See "CBS code options for Primavera XML and Microsoft Project XML" for details. 4. Suffix Mask: Does not apply for Microsoft Project XML files, and this value is ignored when files of this type are imported. You can use a constant or a data element value in the Suffix Mask.

Associate CBS Codes with Activities

In a Project, the activities are grouped together by CBS Codes.

The following shows a sample project data:



In this sample, the activities A1160, A1180 are under a CBS Code built by the following sequence of codes:

- `P -00050-EV-C-OB.5 Execute`
- `P -00050-EV-C-OB.5.ON Onshore`
- `P -00050-EV-C-OB.5.ON.DA LNG Facilities`
- `P -00050-EV-C-OB.5.ON.DA.DA03 LNG Shared Facilities`
- and so forth ...

In this sample, activity A1160 (Plot Plan) is associated with CBS Code: `P -00050-EV-C-OB.5.ON.DA.DA03.F350.U11.095.ABC20.CAct04`. The XML file that is generated by Unifier does not include the entire string of CBS Code with all segments. The structure of the XML file is built based on the Parent/Child CBS segments. As a result, activity A1160 (Plot Plan) has reference to its immediate parent segment, which in this example is `CAct04`, only. The `CAct04` segment will have a reference to `ABC20` segment, the `ABC20` has reference to `095` and so on.

To associate an activity with a CBS Code, Unifier will go through this hierarchical structure of CBS segments to build the final CBS Code. Unifier will perform this function when CBS Code import option is checked on the Options tab of Data Mapping. The CBS Code string generated will then be validated with leaf level CBS Codes created under Cost Manager. If the CBS Code is available, then the activity will be associated with the CBS Code. If the CBS Code is not available, then the import process will fail.

Additional options are available to create entire string of CBS Code. These options can be used to address any structural differences between the way CBS Codes are built in Unifier Cost Manager and CBS Codes in Primavera Project.

- **Levels**: User can choose number of CBS segments that should be considered while processing a Primavera XML file. For example, if user enters 8 then last segment CAct04 will not be considered while building CBS Code.
- **Suffix Mask**: User can optionally enter a suffix mask, which will be added to the end of CBS Codes built based on Primavera XML file. Users can either enter a constant value of select a data element from activity attribute form.

> **Note**: With the help of this data element being selected as suffix, it is possible to build CBS Code and assign it to an activity based on attributes of activities coming from P6 instead of the CBS Code hierarchy of the activities.

DM REST Service (Company Authentication)

The Unifier Document Management Representational State Transfer (REST) Application Program Interface (API) has the following components:

- Folder Services
- Document Services

Notes:

- The UserServices uses the login user access control to run the service calls. The web service uses the company or project admin to run the service calls.
- Before making the REST calls, set the JSESSIONID cookie (with valid value) in the request header (from the REST client code).

In This Section

DM REST API Using Company Authentication..... 161

DM REST API Using Company Authentication

Authorization

Company short name and authentication key is used to authorize these REST calls.

The short name and authentication key will be a part of the form data.

Note: Encode the authentication key in Base64 format when using the REST calls.

Folder Services

Create Folders by Path

POST /ws/rest/dm/folder/create

Purpose and Usage:

Enable users to create folders under a particular folder or under multiple parent folders.

Input:

Form Data

- shortname (String) Required
  company short name.

- authocode (String) Required
  Company authorization key.
- projectnumber (String) Optional.
  Not required for company level.
- data (JSON String) Required
  List of field and values for the folder attribute form.
  - Path
    Specify the full parent path where the folder has to be created.
  - Name
    Name of the folder to create: `Folder1`
  - Path and Name are required fields. Other fields are optional/required based on the attribute form design.

Output:

JSON object containing 'status', 'data', 'message'

Data will be same as input data, with new folder id.

Create Folders by Parent Folder ID

POST /ws/rest/dm/folder/create/<parent_node_id>

Purpose and Usage:

Enable the user to create folders under a specific folder. The parent folder is specified as internal node id in the path parameter.

Input:

Path Parameter

- parent_node_id
  Parent folder node id under which the folders will be created.
  Use the "Get Folder or Document metadata" REST Service to get the node_id value for the parent folder.

Form Data

- shortname (Sting) Required
  Company short name.
- authcode (String) Required
  Company authorization key.
- data (JSON String) Required
  List of field and values for the folder attribute form.
  - Name (Required)

Name of the folder to create. Folder names should be 255 characters or less. Names that contain non-printable ascii, / or \, names with trailing spaces, and the special names "." and ".." are not supported.

- Other fields are optional/required based on the attribute form design.

Output:

JSON object containing 'status', 'data', 'message'

The data will be same as input data with new folder id.

Update Folders Meta Data by Path

POST /ws/rest/dm/folder/update

Purpose and Usage:

Enable the users to update the folder metadata, by using the folder path.

Input:

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authorization key.
- projectnumber (String) Optional
  Project number for Project/Shell level DM. Empty for Company level DM.
- data (JSON String) Required
  List of field and values for the folder attribute form.
  - Path
    Specify the full path of the folder to update.
  - Name
    Name of the folder to perform update.
  - Path and Name are required fields. Other fields are optional.

Output:

JSON object containing 'status', 'data', 'message'

Update Folder Meta Data by Folder ID

POST /ws/rest/dm/folder/update/<node_id>

Purpose and Usage:

Provide the ability to update a specific folder metadata, by using the folder node ID.

Input:

Path Parameter

- node_id
  Node id for the folder which will be updated.
  Use the "Get Folder or Document metadata" REST Service to get the node_id value for the parent folder.

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- data (JSON String) Required
  List of field and values for the folder attribute form.
  - Name
    New name of the folder to perform rename.

Output:

JSON object containing 'status', 'data', 'message'

Get Folders or Documents Meta Data by Path

POST/ws/rest/dm/node/properties

Purpose and Usage:

Enable the system to retrieve folders metadata, or documents metadata, under a particular folder.

Input:

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- projectnumber (String) Optional
  Project number for shell/project level DM. Empty for Company level document manager (DM).

- parentpath (String) Required
  Complete path to a folder in DM.
- nodetype (String) Optional
  Folder if you want only folder properties or "Document" for only document properties. If the no value is specified, then both the folder and document properties will be returned.

Output:

- JSON object containing 'status', 'data', 'message'

Get Folders or Documents Meta Data by Parent Folder ID

POST/ws/rest/dm/node/properties/<parent_folder_id>

Purpose and Usage:

Get the folders or documents metadata by using the parent folder node ID.

Input:

Path Parameter

- parent_node_id
  Parent folder's node ID from which folders or documents meta data will be retrieved

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- nodetype (String) Optional
  "Folder," if you want only the folder properties, or "Document" for the document properties, only. If the no value is specified, then both the folder and document properties will be returned.

Output:

JSON object containing 'status', 'data', 'message'

Document Services

Create Documents by Path

POST /ws/rest/dm/document/create

Purpose and Usage:

- Enables users to add documents (files), and their metadata, to a particular folder.
- Supports adding revision to an existing document.

Input:

Form Data:

This a multi-part request, and all the parameters, must appear before all file parts and in the following order:

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- projectnumber (String) Optional
  Not required for Company level.
- dorevise: (String) Optional
  This flag is used to automatically revise file with same file name. Expected values are "yes" or "no". The default value is, "no."
- data (JSON String) Required
  List of the field and values for the document attribute form.
  - Path
    Full path where the document will be created.The folder path must exists for the document to be created.
  - Name
    Name of the document that needs to be created.

Output:

JSON object containing 'status', 'data', 'message'

Create Documents by Parent Folder ID

POST /ws/rest/dm/document/create/<parent_folder_id>

Purpose and Usage:

- Enables users to add documents under a particular folder, by using the folder node ID.
- Supports adding revision to an existing document.

Input:

Path Parameter:

- parent_folder_id

Parent folder node id under which the documents will be created. Use "Get Folder or Document metadata" REST Service to get the node_id value for the parent folder.

Form Data:

This a multi-part request, and all the parameters, must appear before all file parts and in the following order:

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- dorevise: (String) Optional
  This flag is used to automatically revise file with same file name. Expected values are "yes" or "no". The default value is, "no."
- data (JSON String) Required
  List of field and values for the document attribute form.
  - Name
    Name of the document that needs to be created.

Output:

JSON object containing 'status', 'data', 'message'

Update Documents Meta Data by Path

POST /ws/rest/dm/document/update

Purpose and Usage:

Enables user to modify the attribute form data for documents, under a folder, by using the folder path value.

Input:

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- projectnumber (String) Optional
  Project number for Project/Shell level DM. Empty for Company level DM.
- data (JSON String) Required
  List of the field and values for the document attribute form.
  - Path

Specify the full path of the document (including the document name) to update. Example `/Project Documents/Service docs/Folder1/pipeline estimate.doc`

- Path is a required field. Other fields are optional.

Output:

JSON object containing 'status', 'data', 'message'

Update Document Meta Data by Document ID

POST /ws/rest/dm/document/update/<document_id>

Purpose and Usage:

Enables user to modify the metadata for a particular document, by using the node id.

Input:

Path Parameter:

- document_id

  Document's node id which needs to be updated.

  Use "Get Folder or Document metadata" REST Service to get the node_id value for the document.

Form Data:

- Shortname (String) Required

  Company short name.
- authcode (String) Required

  Company authentication key.
- data (JSON String) Required

  List of the field and values for the document attribute form.

Output:

JSON object containing 'status', 'data', 'message'

Get Document by Path

POST/ws/rest/document

Purpose and Usage:

Provide a web service to retrieve all the documents (files) under a folder, by using the folder path.

Input:

Form Data

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- projectnumber (String) Optional
  Project number for Project/Shell level DM. Empty for Company level DM.
- parentpath (String) Required
  Path to a folder in DM
- iszip (String) Optional
  The value "yes" is to retrieve all the files in a single zip file. The value "no" is to retrieve individual files. The default value is, "yes."

Output:

Output will be a multi-part response.

If iszip option is "yes", then the response file part will be a zip file containing all the documents. If iszip option is "no", then the output will be individual documents in the file part parameter.

Get Documents by Parent Folder ID

POST/ws/rest/dm/document/<parent_folder_id>

Purpose and Usage:

Provide a web service to retrieve all the documents (files) and meta data under a folder, by using the parent folder node ID.

Input:

Path Parameter:

- parent_folder_id
  Parent folder node id under which the documents will be created. Use "Get Folder or Document metadata" REST Service to get the node_id value for the parent folder.

Form Data:

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.
- iszip (String) Optional
  The value "yes" is to retrieve all the files in a single zip file.The value "no" is to retrieve individual files. The default value is, "yes."

Output:

Output will be a multi-part response.

If iszip option is "yes", then the response file part will be a zip file containing all the documents. If iszip option is "no", then the output will be individual documents in the file part parameter.

Get Document by File ID

POST /ws/rest/dm/file/download/<file_id>

Purpose and Usage:

Provide a web service to download the contents of a single document based on file id.

Input:

Path Parameter

- file_id (Required)
  The file id of the document which needs to be downloaded. Use "Get Folder or Document metadata" REST Service to get the file_id value for the document.

Form Data:

- Shortname (String) Required
  Company short name.
- authcode (String) Required
  Company authentication key.

Output:

File stream of the document to be downloaded.

REST Web Services V1

HTTP Methods

GET, POST, PUT, DELETE

Authentication

Pre-requisite for invoking any rest service - Integration user and JWT auth token should be available.

Integration User :

All rest services should be accessed via Integration User.

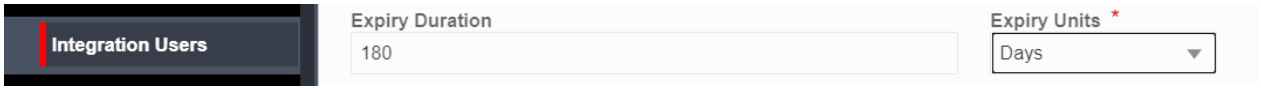
Create an Integration User at Company Workspace → User Administration → Integration User.

Make sure the integration user has the required module permissions enabled.

Integration user token expiration can be set from the Integration user UI. User can create, update Auth Token expiry date to - days, hours and minutes.

1 to 500 is a valid value for Expiry Duration.

By default the value is set to 180 days.



Auth Token :

authentication token is required to access the rest services.

Token is set to be valid for date/time available in the response.

The token can be obtained by the below url.

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

Method - GET

Headers -Authorization:Basic <Basic Auth of integration user>

(To Generate one via Postman rest client - Go to 'Authorization' tab → choose 'Basic Auth' in 'Type' dropdown → Enter Integration user name and password -. Click on 'Update Request'. This generates the necessary Authorization header in 'Headers' tab.)

If the user name/password combination shared in the above Authorization Header is not valid/correct, the service throws **401** status code(Unauthorized)

Other conditions that are checked are - Integration user has to be active.

The expiry date is displayed as per the user date format preferences set in the Integration UI.

Sample response -

{
"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-C6FA7576F55E7D8D9975C46B5805179BD10D890DF15F"
}

Token can now be used in all rest services for authentication mechanism. All rest services should have the below set in the 'Header'.

Header - Sample

Key	Value
Authorization	Bearer eyJ0eXAiOiJEQiJ9.eyJ1c2VybmFtZSI6IiQkZGVsdDMifQ==.02318C44-9F3A-F931-3F14-C6FA7576F55E7D8D9975C46B5805179BD10D890DF15F

Notes:

- Validity of the token is as given in the login. The same can be re-used for subsequent rest requests until the expiry date.
- User can change Expiry Date to minutes, hours and days in UI. Rest will honor those settings and generate new token with given Expiry Duration in the UI.
- If the Authorization token is not valid/correct for subsequent rest requests, those services will throw **401** status code(Unauthorized)
- If the Authorization token is correct but user (for whom token is generated) does not have permission for any rest request, that service will throw **403** status code(Forbidden).
- For every login rest service initiated, a new token is generated, old token is invalidated.

Data

If integration interface is defined for BP then integration form will be used for this service, otherwise all custom DEs defined in the form will be used.

For update, specify only the DEs to be updated.

Data Format

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

Data Transfer

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

Default Integration User

For migrating customers before 19.7, Default integration user with company shortname and authcode with access to existing system services.

Default cloud user has 500 days Auth token validity.

Standards

Get Method -

Will be used to request data from a source when no parameters are sent in the body.

> **Note**: Do not use word "get" in the url.

URL Encoding GET call parameter values

All parameters in GET call must be URL encoded.

For Postman REST client,Use below code in “**pre-request Script**” tab, that will trim extra spaces in params key and encode special characters in params value.

```
pm.request.url.query.all().forEach( (param) =>
{
        param.key = param.key.trim();
        param.value = encodeURIComponent(param.value );
    }
);
```

POST Method -

POST can be used to retrieve data with parameters in body.

Can be used for CREATE, Retrieve data with parameters.

For example getBprecordlist - bpname is mandatory and need to be a sent in request body.

> **Note**: Do not use word "create" in the url.

To distinguish create and get data with parameters term list can be added in the url.

PUT Method -

PUT is used update data.

> **Note**: Do not use word "update" in the url.

Logging:

All REST operations on Unifier gets audited in internal audit log table (not accessible to User, as it is not business case audit logs) .

A background CRON job is created to run on every SUNDAY 4:00AM (server time zone) which will purge older REST internal audit logs which goes beyond 25000 audit rows.

IP Filtering:

Customers have the ability to provide the list of IP addresses which can consumer Unifier REST Webservices(V1 or V2). IP Filtering option is available in Unifier portal in Company Properties - Security Tab.

If the 'IP Filtering Policy' field is checked in company properties, then the remote host will be validated based on their IP Version.

1. If the remote host is IPv4 version then it will be checked against the list of IP addresses provided in the IPv4 text box.
2. If the remote host is of IPv6 version it will be checked against the list of IPs provided in the IPV6 text box.
3. If IP addresses are provided in CIDR format then the remote host IP will be checked against all addresses that come in the range.

In This Section

REST API Details in Document Manager 175
REST API Details in Business Processes 213
REST API Details in Shell Manager 267
REST API Details in Level 281
REST API Details in Space 289
REST API Details in Cost 299
REST API Details in Cashflow 313
REST API Details in Schedule Sheet 445
REST API Details in Exchange Rates 449
REST API Details in Data Structure Setup 453
Partner Company Services V1 503
User Services V1 513
User Defined Report (UDR) Services V1 525
Get Templates List 528
Data 531
Data Format 531
Data Transfer 531
REST API Details in Event Driven Notification 533

REST API Details in Document Manager

In This Section

Folder Services 175
Document Services 185
Search Document or Folder Properties 194
Rename a Node by Node ID 197
Rename Unpublished Document by File ID 198
Get Node Permissions by Node Path and Project Number 199
Get Node Permissions by Node Path, Project Number, and User or Group 202
Add Node Permissions by Node Path and Project Number 204
Update Node Permissions by Node Path and Project Number 207
Remove Node Permissions by Node Path and Project Number 209
Remove Node Permissions by Node Path, Project Number, and User or Group 210
Set Permissions in Document Manager 211

Folder Services

Create Folders by Path

`POST /ws/rest/service/v1/dm/folder/create`

Purpose

This service allows the user to create folders under a parent folder. Supports folder creation under different parent folders.

Input

Form Data (content type : application/x-www-form-urlencoded)

- projectnumber (String) Optional. Not required for company level.
- data (JSON String) Required: List of field and values for the folder attribute form.
  - Path: Specify the full parent path where folder has to created.
  - Name: Name of the folder to create: "Folder1"
  - Path and Name are required fields, other fields are optional/required based on the attribute form design.

Example

To create a folder "REST Folder" under "/Project Documents/Folder112" then data should as below

projectnumber:TestProj-C

data:[{"Path":"/Folder112","Name":"REST Folder","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Comments":"Fold 1" }]

Folder names should be 255 characters or less. Names that will not be supported are those that contain non-printable ascii, / or \, names with trailing spaces, and the special names “.” and “..”.

Output

JSON object containing 'status', 'data', 'message'

Data contains list of fields defined in the folder attribute form. Mandatory to have "Path", the full parent path where folder has to created and "Name" (the name of the folder to create) parameters. Other fields are optional/required based on the attribute form design.

Example:

Successful folder creation:


```
{
"data":
[
{
"Path":"/Folder112/Folder112",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Comments":"Fold 1",
"% Complete":"100",
"Name":"REST Folder",
"node_id":"2003150"
}
],
"message":
[
{
"message":"OK"
}
],
"status":200
}
Failure Condition:
{
"data":
[
{
"Path":"/Folder112/Folder112",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
```

```
"Comments":"Fold 1",
"% Complete":"100",
"Name":"REST Folder"
}
],
"message":
[
{
"message":"Folder with given name already exists"
}
],
"status":1043
}
```


Create Folders by Parent Folder ID

`POST /ws/rest/service/v1/dm/folder/create/<parent_folder_id>`

Purpose

This service allows user to create folders under a specific folder. The parent folder is specified as node id in the service call's path parameter.

Input:

Path Parameter

- parent_folder_id : Parent folder's node id under which the folders will be created.
  - Use “Get Folder or Document meta data by path” REST Service to get the node_id value for the parent folder.

Form Data (Content Type : application/x-www-form-urlencoded)

- data (JSON String) Required: List of field and values for the folder attribute form.
  - Name (Required): Name of the folder to create.
  - Folder names should be 255 characters or less. Names that will not be supported are those that contain non-printable ascii, / or \, names with trailing spaces, and the special names “.” and “..”
  - Other fields are optional/required based on the attribute form design.

Example:

`URL:http://unifier-intg.oracle.com/ws/rest/dm/folder/create/2002896`

To create a folder REST Folder" under "/Project Documents/Service docs" whose node id (2002896) is specified in path parameter then data should as below

data:[{"Name":"REST Folder","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Description":"Fold 1" }]

Output

JSON object containing 'status', 'data', 'message'

Data will be same as input data with new folder id and parent folder id

Example:

Successful folder creation:


```
{
"data":
[
{
"Path":"/Service docs",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"Fold 1",
"% Complete":"100",
"Name":"REST Folder",
"node_id":"2003152"
}
],
"message":
[
{
"message":"OK"
}
],
"status":200
}
```


Failure Condition:


```
{
"data":
[
{
"Path":"/Service docs",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"Fold 1",
"% Complete":"100",
"Name":"REST Folder"
}
],
```

```
"message":
[
{
"message":"Folder with given name already exists"
}
],
"status":1043
}
```

Update Folders Meta Data by Path

`POST /ws/rest/service/v1/dm/folder/update`

Purpose

This service allows user to modify the metadata for documents under a folder using the folder path value.

Input

Form Data (content Type: application/x-www-form-urlencoded)

- projectnumber (String) Optional. Project number for shell/project level DM. Empty for Company level DM.
- data (JSON String) Required: List of field and values for the folder attribute form.
  - Path: Specify the full path of the document to update. Example "/Project Documents/Service docs/Folder1/pipeline estimate.doc"

    Path is required fields, other fields are optional.

Example:

URL: http://<host>:<port>/ws/rest/service/v1/dm/document/update

To update a document "dm1.PNG" under "/test" folder, the data must be:

data:[{"Path":"/test/dm1.PNG","Name":"dm1.PNG","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Description":"dm1.PNG df" }]

Use the GET call to get the structure of the JSON data.

Output

JSON object containing 'status', 'data', 'message'

> **Note**: The read-only fields will not be updated with values from input.

Response in case of success:

```
{
"data":
[
{
```

```
"Path":"/test/dm1.PNG",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"dm1.PNG df",
"% Complete":"100",
"Name":"dm1.PNG"
}
],
"message":
[
{
"message":"OK"
}
],
"status":200
}
```


Response in case of error:


```
{
"data":
[
{
"Path":"/test/dm2.PNG",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"dm1.PNG df",
"% Complete":"100",
"Name":"d1.PNG"
}
],
"message":
[
{
"message":"Invalid folder name."
```

```
}
],
"status":505
}
```

Update Folder Meta Data by Folder ID

POST /ws/rest/service/v1/dm/folder/update/<node_id>

Purpose

This service allows user to update a specific folder meta data using the folder node id.

Input

- Path Parameter
  - node_id: node id of the folder which will be updated.

    Use “Get Folder or Document meta data by path” REST Service to get the node_id value for the folder.
- Form Data (content Type: application/x-www-form-urlencoded)
  - data (JSON String) Required: List of field and values for the folder attribute form.

    Name: New name of the folder to perform rename.

Example:

URL: http://<host>:<port>/ws/rest/service/v1/document/update/298716

To update the folder "Folder1" under "/Project Documents/Service docs" with node id (2003131) is specified in path parameter, the data must be:

data:[{"Name":"Folder1","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Comments":"Fold 2 update" }]

Use the GET call to get the structure of the JSON data.

Output

JSON object containing 'status', 'data', 'message'

> **Note**: The read-only fields will not be updated with input values.

Response in case of Success:

```
{
"data":
[
{
"Path":"/DmPicker",
"Creation Date":"01/01/2016",
"Comments":"Fold 2 update dmpick",
"% Complete":"100",
```

```
"Name":"DmPicker"
}
],
"message":
[
{
"message":"OK"
}
],
"status":200
}
```


Response in case of error:


```
{
"data":
[
{
"Creation Date":"01/01/2016",
"Comments":"Fold 2 update dmpick",
"% Complete":"100",
"Name":"DmPicker"
}
],
"message":
[
{
"message":"Parent path or node id is invalid or does not exists."
}
],
"status":1039
}
```


Get Folders or Documents Meta Data by Path

GET /ws/rest/service/v1/dm/node/properties

Purpose

Provide the ability to retrieve folders or documents meta data under given folder.

Input

Query Parameters

- projectnumber (String) Optional. Project number for shell/project level DM. Empty for Company level DM.

- parentpath (String) Required. Path to a folder in DM
- nodetype (String) Optional : Folder if you want only folder properties or "Document" for only document properties. If no value is specified then both folder and document properties will be returned.

Example:

projectnumber:TestProj-C

parentpath:/u17.1

Output

JSON object containing 'status', 'data', 'message'

"data" will be array of document or folder properties as shown in the below example.


```
{
  "data":
  [
    {
      "Owner":"Company Administrator",
      "projectnumber":null,
      "node_path":"/Company Documents",
      "Description":"",
      "type":"Folder",
      "foreignKeyId":-1000,
      "parentUuid":null,
      "Name":"Company Documents",
      "parent_node_id":285,
      "foreignEntityType":"company",
      "% Complete":"0",
      "Location":"/",
      "node_id":700,
      "upload_date":"2017-12-07T07:54"
    },

    {
      "Revision No.":"",
      "Owner":"Company Administrator",
      "projectnumber":null,
      "node_path":"/Folder
City/Fremont/AutoVue_Applet-Free_Client_TUD.pdf",
      "Description":"",
      "Pub No.":"1",
      "file_name":"AutoVue_Applet-Free_Client_TUD.pdf",
```

```
        "annotate":true,
        "type":"Document",
        "title":"",
        "foreignKeyId":-1000,
        "parentUuid":294,
        "file_size":"448519",
        "uuu_create_by":"Company Administrator",
        "file_id":58,
        "Issue Date":"",
        "foreignEntityType":"company",
        "% Complete":"0",
        "Location":"/Folder City/Fremont",
        "node_id":306,
        "upload_date":"2017-11-03T14:02"
      },

    ],
    "message":
    [
    ],
    "status":200
  }
```


Get Folders or Documents Meta Data by Parent Folder ID

`GET /ws/rest/service/v1/dm/node/properties/<parent_folder_id>`

Purpose

Provide the ability to retrieve the folders or documents meta data under given folder path by using the parent folder node id

Input

Path Parameter

parent_folder_id : Parent folder's node id from which folders/documents meta data will be retrieved.

Query Parameter

nodetype (String) Optional : "Folder" if you want only folder properties or "Document" for only document properties. If no value is specified then both folder and document properties will be returned.

Output

JSON object containing 'status', 'data', 'message'

Example:

```
URL:http://unifier-intg.oracle.com/ws/rest/dm/node/properties/2003148?nodetype=Document
```

Document Services

Create Documents by Path

`POST /ws/rest/service/v1/dm/document/create`

Purpose

- Provide the ability for customers to add documents (files) with their metadata to particular folder.
- Supports adding revision to an existing document.

Input

This a multipart request and All parameters should be BEFORE all file parts and should follow the below order.

- projectnumber (String) Optional. Not required for company level.
- dorevise: (String) Optional .This flag is used to automatically revise file with same file name. Expected values are "yes" or "no". Default is no.
- data (JSON String) Required: List of field and values for the document attribute form.
  - Path: Full path where the document will be created. The folder path must exists for the document to be created.
  - Name: Name of the document that needs to be created.

Example:

To create a document "ProjectDetails.doc" under "/Project Documents/Service docs" then data should as below

[

{"path":"/Service docs","fileName":"ProjectDetails.doc","docTitle":"project details","revNo":"1","issueDate":"06/01/2016", additional document attribute form properties, excluding creation date and owner }

]

Example Multipart Form Request:

> **Note**: The form data body boundary in the example below is "—file upload—", but can be anything (or left unset).

```
-----file upload---
Content-Disposition: form-data; name="shortname"
unifier
-----file upload---
```

```
Content-Disposition: form-data; name="authcode"
MTIzMTIz
-----file upload---
Content-Disposition: form-data; name="dorevise"
yes
-----file upload---
Content-Disposition: form-data; name="projectnumber"
Shell-001
 -----file upload---
Content-Disposition: form-data; name="data"
[
{"Path":"/Service
docs","Name":"ProjectDetails.doc","docTitle":"project
details","revNo":"1","issueDate":"06/01/2016", additional document
attribute form properties, excluding creation date and owner }
]
-----file upload---
Content-Disposition: form-data; name="0"; filename="b.txt"
Content-Type: text/plain
test document contents
-----file upload-----
```

Output

Json object containing 'status', 'data', 'message'

Create Documents by Parent Folder ID

POST /ws/rest/service/v1/dm/document/create/<parent_folder_id>

Purpose

- Provide the ability for customers to add documents (files) to a particular folder using the folder node id.
- Supports adding revision to an existing document.

Input

Path Parameter

- parent_folder_id : Parent folder's node id under which the documents will be created.
  - Use “Get Folder or Document meta data by path” REST Service to get the node_id value for the folder.

This a multipart request and All parameters should be BEFORE all file parts and should follow the below order.

- projectnumber (String) Optional.

Project number for shell/project level DM. Empty for Company level DM. (If projectnumber is not given, then the document will be created under company workspace. If the parent folder Id is not under company workspace, then the document will not be visible even in company workspace).

- dorevise: (String) Optional.

  This flag is used to automatically revise file with same file name. Expected values are "yes" or "no". Default is no.

- data (JSON String) Required:

  List of field and values for the document attribute form.

  - Name: Name of the document that needs to be created.

Example:

```
To create a document "ProjectDetails.doc" under "/Project
Documents/Service docs" then data should as below
[
 {"fileName":"ProjectDetails.doc","docTitle":"project
details","revNo":"1","issueDate":"06/01/2016", additional document
attribute form properties, excluding creation date and owner }
]
```

Example Multipart Form Request:

> **Note**: The form data body boundary in the example below is "—file upload—", but can be anything (or left unset).

```
-----file upload---
Content-Disposition: form-data; name="shortname"
unifier
-----file upload---
Content-Disposition: form-data; name="authcode"
MTIzMTIz
-----file upload---
Content-Disposition: form-data; name="dorevise"  yes
 -----file upload---
Content-Disposition: form-data; name="data"
[
 {"Name":"ProjectDetails.doc","Owner":"Company
Administrator","Creation Date":"01/01/2016","%
Complete":"100","Description":"Fold 1" }
]
-----file upload---
Content-Disposition: form-data; name="0"; filename="b.txt"
Content-Type: text/plain
test document contents
```

```
-----file upload-----
```

Output

Json object containing 'status', 'data', 'message'

Update Documents Meta Data by Path

`POST /ws/rest/service/v1/dm/document/update`

Purpose

Provide the ability to modify the metadata for documents under a folder using the folder path value.

Input

Form Data (content Type: application/x-www-form-urlencoded)

- projectnumber (String) Optional. Project number for shell/project level DM. Empty for Company level DM.
- data (JSON String) Required: List of field and values for the document attribute form.
  - Path: Specify the full path of the document to update. Example "/Project Documents/Service docs/Folder1/pipeline estimate.doc"

Path is a required field. Other fields are optional.

Example:

URL: http://<host>:<port>/ws/rest/service/v1/dm/document/update

To update a document "dm1.PNG" under "/test" folder the data should must be:

data:[{"Path":"/test/dm1.PNG","Name":"dm1.PNG","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Description":"dm1.PNG df" }]

Use the GET call to get the structure of the JSON data.

Output:

JSON object containing 'status', 'data', 'message'

> **Note**: The read-only fields will not be updated with values from input.

Response in case of success:

```
{
"data":
[
{
"Path":"/test/dm1.PNG",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"dm1.PNG df",
"% Complete":"100",
```

```
"Name":"dm1.PNG"
}
],
"message":
[
{
"message":"OK"
}
],
"status":200
}
```


Response in case of error:


```
{
"data":
[
{
"Path":"/test/dm2.PNG",
"Creation Date":"01/01/2016",
"Owner":"Company Administrator",
"Description":"dm1.PNG df",
"% Complete":"100",
"Name":"d1.PNG"
}
],
"message":
[
{
"message":"Invalid folder name."
}
],
"status":505
}
```


Update Document Meta Data by Document ID

POST /ws/rest/service/v1/dm/document/update/<document_id>

Purpose

Provide the ability to modify the metadata for a particular document using the node id.

Input

Path Parameter

- document_id : Node id of the document which needs to be updated.
  - Use “Get Folder or Document meta data by path” REST Service to get the node_id value for the document.

Form Data (content Type: application/x-www-form-urlencoded)

- data (JSON String) Required: List of field and values for the document attribute form.

Example:

URL: http://<host>:<port>/ws/rest/service/v1/dm/document/update/298717

To update a document "dm1.PNG" under ""/test" the data must be:

projectnumber:B-001

data:[{"Path":"/test/dm1.PNG","Name":"dm1.PNG","Owner":"Company Administrator","Creation Date":"01/01/2016","% Complete":"100","Description":"dm1.PNG data" }]

Use the GET call to get the structure of the JSON data.

Output

JSON object containing 'status', 'data', 'message'

Response in case of success:

```
{ "data": [ { "Path":"/test/dm1.PNG", "Creation Date":"01/01/2016",
"Owner":"Company Administrator", "Description":"dm1.PNG data", "%
Complete":"100", "Name":"dm1.PNG" } ], "message": [ { "message":"OK" }
], "status":200 }
```

Response in case of error:

```
{ "data": [ { "Path":"/test/dm1.PNG", "Creation Date":"01/01/2016",
"Owner":"Company Administrator", "Description":"dm1.PNG data", "%
Complete":"100", "Name":"dm1.PNG" } ], "message": [ { "message":"Parent
path or node id is invalid or does not exists." } ], "status":1039 }
```

Get Documents by Path

`GET /ws/rest/service/v1/dm/document`

Purpose

Provide a web service to retrieve all the documents (files) and meta data under a folder using the folder path.

Input

Query Parameters

- projectnumber (String) Optional. Project number for shell/project level DM. Empty for Company level DM.
- parentpath (String) Required. Path to a folder in DM

- iszip (String) Optional: "yes" to get all the files in a single zip file. "no" to get individual files. Default is ''

Output

Output will be a multi-part response.

- If iszip option is yes then the response file part will be a zip file containing all the documents.
- If iszip option is no then the output will be individual documents in the file part parameter.

Example:

There are 2 files b.txt (content is test test) and ucm_migration_issue.txt (content is 1. Most of the cases the wrong file referenced is a report) under the selected directory.

iszip = no

Response Header

Content-Type = multipart/form-data; boundary=QXL6HDrSIJF1imDRjdgUC8bTbHIq1z4PUZ

Response Body

--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye

Content-Disposition: form-data; name="response"

Content-Type: application/octet-stream

Content-Transfer-Encoding: 8bit

{"status":200,"data":[],"message": ""}

--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye

Content-Disposition: form-data; name="b.txt"; filename="b.txt"

Content-Type: application/octet-stream

Content-Transfer-Encoding: binary

test test

--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye

Content-Disposition: form-data; name="ucm_migration_issue.txt"; filename="ucm_migration_issue.txt"

Content-Type: application/octet-stream

Content-Transfer-Encoding: binary

1. Most of the cases the wrong file referenced is a report

--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye--

{ "data": [ ], "message": [ ], "status":200 }

iszip = yes

Response Header

Content-Type = application/octet-stream

Response Body

zip file binary data

In case of error:

Output:

{

```
"data":
[
],
"message":
[
{
"message":"Parent path or node id is invalid or does not exists."
}
],
"status":1039
}
```

Get Documents by Parent Folder ID

GET /ws/rest/service/v1/dm/document/<parent_node_id>

Purpose

Provide a web service to retrieve all the documents (files) and meta data under a folder using the parent folder node id.

Input

Path Parameter

- parent_folder_id : Parent folder's node id from which to get the documents.
  - Use “Get Folder or Document meta data by path” REST Service to get the node_id value for the folder.

Query Parameter

- iszip (String) Optional: "yes" to get all the files in a single zip file. "no" to get individual files. Default is "yes"

Output

Output will be a multi-part response.

- If iszip option is yes then the response file part will be a zip file containing all the documents.
- If iszip option is no then the output will be individual documents in the file part parameter.

Example:

To get the documents under parent_node_id 55889 request URL should be as below:

URL: http://<host>:<port>/ws/rest/service/v1/dm/document/55889

There are 2 files b.txt (content is test test) and ucm_migration_issue.txt (content is 1. Most of the cases the wrong file referenced is a report) under the selected directory.

iszip = no

Response Header

Content-Type = multipart/form-data; boundary=QXL6HDrSIJF1imDRjdgUC8bTbHIq1z4PUZ

Response Body

--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye
Content-Disposition: form-data; name="response"
Content-Type: application/octet-stream
Content-Transfer-Encoding: 8bit
{"status":200,"data":[],"message": ""}
--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye
Content-Disposition: form-data; name="b.txt"; filename="b.txt"
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
test test
--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye
Content-Disposition: form-data; name="ucm_migration_issue.txt";
filename="ucm_migration_issue.txt"
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
1. Most of the cases the wrong file referenced is a report
--5MJDL8Wb3f7FtknE7iXUzMYy-W1KqjuFfd6bye--
{ "data": [ ], "message": [ ], "status":200 }

iszip = yes

Response Header

Content-Type = application/octet-stream

Response Body

zip file binary data

In case of error:

Output:

```
{
"data":
[
],
"message":
[
{
"message":"Parent path or node id is invalid or does not exists."
}
],
"status":1039
}
```

Get Document as Tiff File

```
GET /ws/rest/service/v1/dm/file/view/<file_id>
```

Purpose

Provide a web service to retrieve a file as a multi page tiff file.

Input

Path Parameter: file_id: internal file id

Output

Request will return a tiff file

Get Document by File ID

`GET /ws/rest/service/v1/dm//file/download/<file_id>`

Purpose

Provide a web service to download the contents of a single document based on file id.

Input

Path Parameter

- file_id(Required) : The file id which needs to be downloaded
  - Use “Get Folder or Document meta data by path” REST Service to get the file_id value for the document.

Output

Document file stream object.

Example:

```
HTTP/1.1 200 OK
Date: Thu, 01 Sep 2016 15:35:07 GMT
Transfer-Encoding: chunked
Content-Type: application/octet-stream
Content-Disposition: attachment; filename=Autovue_reply.txt
<< Contents of the file being downloaded >>
```

Search Document or Folder Properties

`POST /ws/rest/service/v1/dm/search`

Purpose

Provide a web service to search a document/folder properties.

Input

Form Data (content Type: application/x-www-form-urlencoded)

- projectnumber (String) Optional. Project number for shell/project level DM. Empty for Company level DM.
- parentpath (String) Required. Path to a folder in DM

- match (String) : Search condition to apply for search terms. Default is AND. Possible values AND / OR
- nodetype (String) Optional. Specify if you want to search in "folders" or "files". Default is to search in both folders and documents. Values can be "Folder" or "Document".
- query (JSON String) Required if simplequery is not specified : array of search terms. Each of the search term will be joined with AND condition.
  - name: uDesinger element name to be searched. (Required)
  - comparator: Operation to be performed on the value (Optional)
  - value: Search value (Required)
  - value2: Search value for the range comparator (Optional)

  Example:

  ```
  "query":[{ "name":"uuu_file_title", "label":"Title","value":"first
  document title", "comparator":"like" }, {
  "name":"uuu_file_issue_date", "label":"Issue Date",
  "value":"2017-12-07T07:54", "value2":"2017-20-07T07:54",
  "comparator":"range" } ]
  ```

  The above request will search for documents with title like "first document title" and issue date between 2017-12-07T07:54 and 2017-20-07T07:54.

Name	Label
uuu_dm_node_name	Name
uuu_dm_create_by	Owner
uuu_file_title	Title

Comparator:

- like : For sting data types. Will perform a like query, %value% (Default)
- eq : For string and numeric data type. Will perform extact match.
- lt : Numeric , Date values. Will perform less than value.
- gt : Numeric , Date values. Will perform greater than value.
- le : Numeric , Date values. Will perform less than or equal value.
- ge : Numeric , Date values. Will perform greater than or equal value.
- ne : Numeric , Date values. Will perform not equal value.
- range : Numeric , Date values. Will perform range search between value and value2

Example:

```
"projectnumber":"0002"
"parentpath":"/Meeting%20Minutes"
"query":[ { "name":"uuu_dm_node_name",
"label":"Name","value":"meetingnotes.txt", "comparator":"like" } ,[ {
"name":"", "label":"Pub No.", "value":"2", "comparator":"gt" }]
"match":"OR"
```

Output

Output will be same as /dm/node/properties REST call.

Json object containing 'status', 'data', 'message'

"data" will be array of document or folder properties matching the search term specified in the request.

Example:


```
{
  "data":
  [
    {
      "Owner":"Company Administrator",
      "projectnumber":null,
      "node_path":"/Company Documents",
      "Description":"",
      "type":"Folder",
      "foreignKeyId":-1000,
      "parentUuid":null,
      "Name":"Company Documents",
      "parent_node_id":285,
      "foreignEntityType":"company",
      "% Complete":"0",
      "Location":"/",
      "node_id":700,
      "upload_date":"2017-12-07T07:54"
    },

    {
      "Revision No.":"",
      "Owner":"Company Administrator",
      "projectnumber":null,
      "node_path":"/Folder
City/Fremont/AutoVue_Applet-Free_Client_TUD.pdf",
      "Description":"",
      "Pub No.":"1",
      "file_name":"AutoVue_Applet-Free_Client_TUD.pdf",
      "annotate":true,
      "type":"Document",
      "title":"",
      "foreignKeyId":-1000,
      "parentUuid":294,
```

```
        "file_size":"448519",
        "uuu_create_by":"Company Administrator",
        "file_id":58,
        "Issue Date":"",
        "foreignEntityType":"company",
        "% Complete":"0",
        "Location":"/Folder City/Fremont",
        "node_id":306,
        "upload_date":"2017-11-03T14:02"
      },

    ],
    "message":
    [
    ],
    "status":200
  }
```


Rename a Node by Node ID

POST /ws/rest/service/v1/dm/node/rename/{node_id}

Purpose

Provide a web service to rename a node(folder/document) based on node id.

Input

Path Parameter

- node_id(Required) : The node id which needs to be renamed.

Form Data (content Type: application/x-www-form-urlencoded)

- data: `[{"new_node_name":"rename.json","forceful":"yes" }]`

Output

JSON object containing 'status', 'data', 'message'

Example:

To rename a folder/document with node_id 86375, give the URL as below:
URL: http://<host>:<port>/ws/rest/service/v1/dm/node/rename/86375
data:[{"new_node_name":"rename89.pdf","forceful":"yes" }] projectnumber:B001

Response in case success:

{

"data": [{

"Path": "/RESTMAINFOLDER/Auto_Fold1",
"forceful": "yes",
"node_name_input": "rename89.pdf",
"Name": "rename89.pdf"
}],
"message": [{"message": "Renamed node successfully"}],
"status": 200
}

Response in case of error:

{
"data": [{
"Path": "/RESTMAINFOLDER/Auto_Fold1",
"forceful": "yes",
"node_name_input": "rename.pdf",
"Name": "rename.pdf"
}],
"message": [{"message": "New node name and old node name are same"}],
"status": 505
}

Rename Unpublished Document by File ID

POST /ws/rest/service/v1/dm/node/rename/unpublished/{file_id}

Purpose

Provide a web service to rename an unpublished document based on file id.

Input

Path Parameter

- file_id(Required) : The unpublished document file_id which needs to be renamed.
- Form Data (content Type: application/x-www-form-urlencoded)
  - data: `[{"new_node_name":"rename.json","forceful":"yes" }]`

Output

JSON object containing 'status', 'data', 'message'

Example:

To rename an unpublished document with file_id 219 give URL as below:

URL: http://<host>:<port>/ws/rest/service/v1/dm/node/rename/unpublished/219
data: [{"new_node_name":"renamefile.dwg","forceful":"yes" }]

Response in case of success:

```
{
"data": [ {
"Path": "",
"forceful": "yes",
"node_name_input": "renamefile.dwg",
"Name": "renamefile(1).dwg"
}],
"message": [{"message": "Renamed unpublished document successfully"}],
"status": 200
}
```

Response in case of error:

```
{
"data": [ {
"Path": "",
"forceful": "yes",
"node_name_input": "rename.dwg",
"Name": "rename.dwg"
}],
"message": [{"message": "New file name and old file name are same"}],
"status": 505
}
}
```

Get Node Permissions by Node Path and Project Number

`GET /ws/rest/service/v1/dm/permission/{projectNumber}/node/{node_path}`

Purpose

This service allows user to retrieve permissions of a node by node_path and project number.

Input

Path parameter:

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path: (Path parameter) full node_path of the required node.

Output:

JSON object containing permissions data

Example:

To get the permissions of node "test3" under path "/tes2" of projectnumber b1, give the url as follows.

http://<host>:port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3

Response in case of success:

{
"Inheritance": false,
"ApplyToAllSubFolders": false,
"UserPermission": [
{
"Type": "PG",
"GroupName": "Asset Managers",
"FullName": "Asset Managers",
"DocumentPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"Download": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"AddComments": 1,
"Revise": 1
},
"FolderPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1
},

{
"Type": "U",
"LoginName": "coadmin",
"FullName": "Company Administrator",
"DocumentPermission": {
"View": 1,
"Move": 0,
"Copy": 0,
"Delete": 0,
"Download": 0,
"ModifyProperties": 0,
"ModifyPermissions": 0,
"AddComments": 0,
"Revise": 0
},
"FolderPermission": {
"View": 1,
"Move": 0,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1
}
}
]
}

Response in case of error:

{
"message": ["Invalid Node Path"],
"status": 1075
}

Get Node Permissions by Node Path, Project Number, and User or Group

```
GET
/ws/rest/service/v1/dm/permission/{projectNumber}/node/{node_path}/user
/{username}?usertype={usertype}
```

Purpose:

This service allows user to retrieve permissions of a node by node_path, project number, user/group name and userType(either 'U' for user, 'PG' for group)

Input:

Path parameters:

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path : full node_path of the required node.

username: user/group name. If it is user (user login name should be given, if its group, group name should be given)

usertype: QueryParameter: 'U' for user, 'PG' for group.

POST body is JSON

Request body:

{

"Inheritance": false,

"ApplyToAllSubFolders": false,

"UserPermission": [

{

"Type": "PG",

"GroupName": "Asset Managers",

"FullName": "Asset Managers",

"DocumentPermission": {

"View": 1,

"Move": 1,

"Copy": 1,

"Delete": 1,

"Download": 1,

"ModifyProperties": 1,

"ModifyPermissions": 1,

"AddComments": 1,

"Revise": 1

},

"FolderPermission": {

"View": 1,

"Move": 1,

"Copy": 1,

"Delete": 1,

"ModifyProperties": 1,

"ModifyPermissions": 1,

"CreateSubFolders": 1,

"AddDocuments": 1

}

}

]

}

Output:

JSON object containing permissions data

Example:

To get the permissions of node "test3" under path "/test2" of projectnumber b1 for a given group "Asset Managers", give the url as follows.

http://<host>:port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3/user/Asset%20Managers?usertype=PG

usertype is either 'U' for user or 'PG' for project group.

Response in case of success:

```
"Inheritance": false,
"ApplyToAllSubFolders": false,
"UserPermission": [
{
"Type": "PG",
"GroupName": "Asset Managers",
"FullName": "Asset Managers",
"DocumentPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
```

```
"Download": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"AddComments": 1,
"Revise": 1
},
"FolderPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1}}]}
```

Response in case of error:

```
{
"message": ["Invalid user Type. Type is mandatory to identify either a
user or a group"],
"status": 1075
}
```

Add Node Permissions by Node Path and Project Number

`POST /ws/rest/service/v1 /dm/permission/{projectNumber}/node/{node_path}`

Purpose

This service provides the ability to add/update permissions of a node.

Input

Path parameters:

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path : full node_path of the required node.

POST body is a JSON

Request body:

{

"Inheritance": false,

"ApplyToAllSubFolders": false,

"UserPermission": [

{

"Type": "PG",
"GroupName": "Asset Managers",
"FullName": "Asset Managers",
"DocumentPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"Download": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"AddComments": 1,
"Revise": 1
},
"FolderPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1
}
}
]
}

Output:

JSON object containing 'status', 'data', 'message'

Example:

To add permission of a node test3 under node path "/test2" of project b1 give input as follows:

URL: http://<host>:<port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3

Request body:

{
"Inheritance": false,
"ApplyToAllSubFolders": false,
"UserPermission": [
{
"Type": "PG",
"GroupName": "Asset Managers",
"FullName": "Asset Managers",
"DocumentPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"Download": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"AddComments": 1,
"Revise": 1
},
"FolderPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1
}
}
]
}

Response in case of success:

[{
"message":
["given permissions are added successfully"
],
"status":200
}]

Response in case of error:

[{
"message":
["Invalid user name or user does not have permission to the current project"
],
"status":1076
}]

Update Node Permissions by Node Path and Project Number

PUT /ws/rest/service/v1 /dm/permission/{projectNumber}/node/{node_path}

Purpose

This service provides the ability to add/update permissions of a node.

Input

Path parameters:

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path : full node_path of the required node.

Output:

JSON object containing 'status', 'data', 'message'

Example:

To update permission of a node test3 under node path /test2 of project b1, give input as follows:

URL: http://<host>:<port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3

Request Body:

{
"Inheritance": false,
"ApplyToAllSubFolders": false,
"UserPermission": [

{
"Type": "PG",
"GroupName": "Asset Managers",
"FullName": "Asset Managers",
"DocumentPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"Download": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"AddComments": 1,
"Revise": 1
},
"FolderPermission": {
"View": 1,
"Move": 1,
"Copy": 1,
"Delete": 1,
"ModifyProperties": 1,
"ModifyPermissions": 1,
"CreateSubFolders": 1,
"AddDocuments": 1
}
}
]
}

Response in case success:

[{
"message":
["node permissions are updated successfully"
],

"status":200

}]

Response in case error:

[{ "message": ["Invalid user name or user does not have permission to the current project"], "status":1076 }]]

Remove Node Permissions by Node Path and Project Number

`PUT /ws/rest/service/v1/dm/permission/{projectNumber}/node/{node_path}/remove`

Purpose

This service allows user to retrieve permissions of a node by node_path and project number.

Input

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path : full node_path of the required node.

Output:

Status of the remove permission.

Example:

To remove the permissions of node test3 under "/test2" of projectnumber b1, give the URL as follows.

URL http://<host>:port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3

Response in case of success:

```
[
{
"message":
[ "all permissions removed successfully for this node"
],
"status":200
}]]
```

Response in case of error:

```
[
{
"message":
[ "The given node path does not exist"
],
"status":1074
}]]
```

Remove Node Permissions by Node Path, Project Number, and User or Group

```
PUT
/ws/rest/service/v1/dm/permission/{projectNumber}/node/{node_path}/remo
ve/user/{username}?usertype={usertype}
```

Purpose

This service allows user to retrieve permissions of a node by node_path, project number, user/group name and user Type (either 'U' for user, 'PG' for group)

Input

projectnumber: (Optional) project number(if not given, default is company workspace)

node_path : full node_path of the required node.

username: comma separated user/group names based on usertype. If it is user(user login name should be given, if its group, group name should be given)

usertype: QueryParameter: 'U' for user, 'PG' for group.

Output:

Status of the remove permission.

Example:

To remove permissions of node "test3" under node path /test2 of projectnumber b1 for a given group "Asset Managers", give the URL as follows.

http://<host>:port>/ws/rest/service/v1/dm/permission/b1/node/test2/test3/user/Asset%20Managers?usertype=PG

> **Note**: the usertype is either 'U' for user or 'PG' for project group.

Response in case of success:

[{ "message":

["Permissions removed successfully"

],

"status":200

}]]

Response in case of error: if given node path does not exist in the existing permission list:

[{"message":

["The given node path does not exist"

],

"status":1074

}]]

Set Permissions in Document Manager

You can set permissions for files and folders in the Document Manager (DM) by way of the restful web services.

Setting permissions to folders and documents in the DM to users or groups through REST Services involves one of the following:

- Granting explicit permissions (calling out a file or folder, specifying user or group name, and granting permissions to those users or groups)
- Inheriting permissions from a parent folder into child files or folders
- Applying (pushing) permissions from a parent folder to sub-folders or files
- Remove users or groups from having permissions to files or folders in the DM through REST Services
- Get permissions on files or folders using REST Services

Authorization

Every V1 REST services includes JWT token in the Authorization header. The integration username and password which can be created in Unifier under Company Workspaces > User Administration > Integration User are required to create this token. Make a GET request to "`/ws/rest/service/v1/login`" service with integration username and password as Basic authorization.

If the call is successful with 200 response code, then the response will contain the JWT token and the expiry date (The token is valid for 6 months from issuance).

The 401 status code will be returned when the username/password combination is incorrect, the User is not an integration user, or when the User is not active.

Sample response

```
{
"expiryDate": "2019/01/10 12:00 AM",
"status": 200,
"token":
"eyJ0eXAiOiJEQiJ9.eyJ1c2VybmFtZSI6IiQkaW50ZWdyYXRpb25icCJ9.2F17ECA4-F553-DF21-BD3F-389AE3680675C80772B0EBF7BAD220BF36BA611BB6A5"
}
```

All Rest Services must have the following set in the Header.

Header Sample

Key	Value
Authorization	Bearer eyJ0eXAiOiJEQiJ9.eyJ1c2VybmFtZSI6IiQkaW50ZWdyYXRpb25icCJ9.2F17ECA4-F553-DF21-BD3F-389AE3680675C80772B0EBF7BAD220BF36BA611BB6A5

> **Note**: The same authorization token can be re-used for subsequent Rest Service requests, until the expiry date.

Granting, Inheriting, and Pushing permissions to files or folders

Users can use REST Services to call out a file or folder in the DM (Company, Shell, or Project), specify names of the users or groups at that location, and *grant* any (or all) of the existing permissions (users or groups) available in the UI for that file or folder.

Users can grant permissions to a file or folder in REST Services by stating that a file or folder "Inherit permissions from the parent folder."

If a user states that a file or folder "Inherit permissions from the parent folder" and also calls user or group names and grants permissions to those individually, then the "Inherit permissions from the parent folder" setting will take precedence and the manually called out permissions will be ignored.

Users can call out a folder and grant permissions to files or folders located within the folder by stating in the REST Services: "Apply these permissions to documents and sub-folders."

Removing Permissions to Files or Folders

Users can use REST Services to call out a file or folder in the DM (Company, Shell, or Project), specify names of the users or groups at that location, and *delete* any (or all) of the existing permissions (users or groups) available in the UI for that file or folder.

You can remove users or groups from permissions through REST Services only if the **Inherit permissions from the parent folder** option is not selected. If this option is selected, then the remove of users or groups will not go through. You can change the inherit permissions condition (on a file or folder by) by deselecting the **Inherit permissions from the parent folder** through REST Services.

Get Permissions on Files or Folders

You can use REST Services to get permissions on a file or folder located in the DM.

REST API Details in Business Processes

Ensure that the integration user has the required Business Process Permissions.

In This Section

Fetch BP Record List 213
Get BP Record 221
Get BP Record With Attachments 224
Create BP Record 232
Update BP Record 240
Update BP Record with Attachment (REST API Details in Business Processes) ... 247
Create BP Record with Attachment 254
Fetch BP Record List with filter_criteria 262

Fetch BP Record List

POST /ws/rest/service/v1/bp/records/{project_number}

Purpose:

Get all list of record in of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

filter_condition

```
{
    "bpname" : "Vendors",
    "lineitem" : "yes",
    "lineitem_fields" : " uuu_tab_id;title",
    "filter_condition" : "status=Active",
    "record_fields" : "uuu_user_id;uuu_record_last_update_date"
}
```

POST body is a JSON, POST call has input & output both as JSON in the body.

The "bpname" is mandatory.

The "record_fields" (optional) is list of fields. That is to state: The data element names in the record upper form , only those fields (along with record_no) will be listed out in result.

The "lineitem_fields"(optional) is list of fields in the record lineitem form (only those fields will be listed out in the result).

The "lineitem" values "yes' or "no", default "yes," if not provided. If "no," then the record lineitem details will not be fetched in the result.

The "filter_condition" (optional) is the condition on the record level. The records which satisfy that criteria will be fetched. Only the "=" (equal) condition is supported.

Filter Criteria

BP Record list can be filtered further using filter_criteria,

- Text, String = Pattern matching
- Number, Cost = Ranges and arithmetic / logical operators
- Date = Ranges

"filter_criteria" should be added in POST data.

filter_criteria

```
{
    "bpname" : "Vendors",
    "lineitem" : "yes",
    "lineitem_fields" : "uuu_tab_id;title",
    "record_fields" : "uuu_user_id;uuu_record_last_update_date",

    "filter_criteria":{
        "join":"OR",
        "filter":[
            {
                "field":"record_no",
                "value":"VEN",
                "condition_type":"like"
            },
            {
                "field":"status",
                "value":"Active",
                "condition_type":"eq"
            },
            {
                "field":"uuu_file_title",
                "value":"first document title",
```

```
                "condition_type":"like"
            },
            {
                "field":"uuu_file_issue_date",
                "value":"12-07-2017 07:54:00",
                "value2":"05-07-2018",
                "condition_type":"range"
            }
        ]
    }
}
```

filter_criteria key Definitions:

1) The "join": (optional) Specify if the filter condition has to be combined using OR or AND operator. Default is AND.
2) The "filter": List of filter conditions. Each condition will have "field", "value" ("value2" if condition_type is range) and condition_type.
   - The "field": BP form DE name for which the filter condition has to be applied.
   - The "value": value for the DE to filter the data.
   - The "value2": second value for the range condition. Not required to be specified for other conditions.
   - The "condition_type": the filter condition that needs to be applied.

Possible value for condition_type:

condition_type	Notes	Supported Data Element Types
like	For string data types. The value can contain the SQL wildcard characters when like is specified.	Text (String)
eq	For string, numeric and date data type. Will perform the exact match.	Text (String), Number, Cost, Date
lt	For numeric values. Will perform less than value.	Number, Cost, Date
gt	For numeric values. Will perform greater than value.	Number, Cost, Date
lteq	For numeric values. Will perform less than or equal value.	Number, Cost, Date
gteq	For numeric values. Will perform greater than or equal value.	Number, Cost, Date

neq	For numeric values. Will perform not equal value.	Text (String), Number, Cost, Date
range	Numeric, Date values. Will perform range search between value and value2.	Number, Cost, Date

> **Note**: The **Input Date** fields should be provided in "MM-dd-yyyy HH:mm:ss" or ""MM-dd-yyyy" format in **filter_criteria**.

Earlier "filter_condition" (optional) is still supported, and it is the condition on the record level, records which satisfy that criteria will be fetched.

If both "filter_condition" and "filter_criteria" are provided together then It will be "AND" condition between "filter_condition" and "filter_criteria".

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 otherwise it will be "success".

The below funding details for general spends, payment application, or generic cost BPs will be present in the response

- unassigned_amount
- fund (JSON array containing below)
  - code
  - fund_balance
  - assigned_amount

> **Note**: If an integration interface is defined for the BP, the all of the fields defined in the interface will be sent in the response. If an integration interface is not defined, then all of the custom-defined fields will be sent in response.

Get BP list Sample Response


```
{
  "data":
  [
    {
      "uuu_user_id":"a@abc.com",
      "uuu_record_last_update_date":"2018/04/23 11:06",
      "_bp_lineitems":
      [
        {
          "uuu_tab_id":"List of Contacts",
```

```
          "li_num":1,
          "title":"t"
        }
      ],
      "record_no":"VEN-0023"
    },
    {
      "uuu_user_id":"a@abc.com",
      "uuu_record_last_update_date":"2018/05/28 04:34",
      "_bp_lineitems":
      [
        {
          "uuu_tab_id":"List of Contacts",
          "li_num":1,
          "title":"t"
        },
        {
          "uuu_tab_id":"List of Contacts",
          "li_num":2,
          "title":"t"
        }
      ],
      "record_no":"VEN-0024"
    }
  ],
  "message":
  [ "success"
  ],
  "status":200
}
```


Sample response with fund details


```
// For lineitem and CBS assignment rule
{
```

```
"data": [
  {
    "amount": 140,
    "_bp_lineitems": [
      {
        "uuu_quantity": 1,
        "amount": 10,
        "group_id": 0,
        "bItemID": "1_D_D~~1_D_D",
        "uuu_tab_id": "Standard",
        "uuu_unit_price": 10,
        "short_desc": "a1",
        "li_num": 1,
        "unassigned_amount":20.30
        "funds": [
          {
            "code": "City Funding",
            "fund_balance": 790,
            "assigned_amount":20
          },
          {
            "code": "State Funding",
            "fund_balance": 3500,
            "assigned_amount":560
          }
        ]
      },
      {
        "uuu_quantity": 2,
        "amount": 40,
        "group_id": 0,
        "bItemID": "2_D_D~~2_D_D",
        "uuu_tab_id": "Standard",
        "uuu_unit_price": 20,
```

```
          "short_desc": "a2",
          "li_num": 2
        },
        {
          "uuu_quantity": 3,
          "amount": 90,
          "group_id": 0,
          "bItemID": "1_D_D~~1_D_D",
          "uuu_tab_id": "Standard",
          "uuu_unit_price": 30,
          "short_desc": "a3",
          "li_num": 3
        }
      ],
      "record_no": "uxcost5-0010",
      "due_date": null,
      "title": "multiple li 2",
      "status": "Pending"
    }
  ],
  "message": [
    "success"
  ],
  "status": 200
}

//For record level assignment rule
{
  "data": [
    {
      "amount": 51600,
      "unassigned_amount":7000,
      "funds": [
```

```
    {
      "code": "City Funding",
     "fund_balance": 3500,
     "assigned_amount":560
    },
    {
      "code": "State Funding",
      "fund_balance": 3500,
      "assigned_amount":560
    }
  ],
  "_bp_lineitems": [
    {
      "uuu_quantity": 50,
      "amount": 50000,
      "uuu_effective_date": null,
      "group_id": 0,
      "bItemID": "1_D_D~~1_D_D",
      "uuu_tab_id": "Standard",
      "upaItemUnitPrice": 1000,
      "short_desc": "short desc 1",
      "li_num": 1
    },
    {
      "uuu_quantity": 4,
      "amount": 1600,
      "uuu_effective_date": null,
      "group_id": 0,
      "bItemID": "1_D_D~~1_D_D",
      "uuu_tab_id": "Standard",
      "upaItemUnitPrice": 400,
      "short_desc": "short desc 2",
      "li_num": 2
    }
```

```
      ],
      "record_no": "uxcost4-0004",
      "due_date": null,
      "refid": "uxcost1-0001",
      "title": null,
      "status": "Approved"
    }
  ],
  "message": [
    "success"
  ],
  "status": 200
}
```

Get BP Record

GET /ws/rest/service/v1/bp/record/{project_number}

Purpose:

Get a record in of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_num: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

GET URL Parameter

```
input =
 {
    "bpname" : "Vendors",
    "record_no" : "VEN-0023",
    "lineitem":"yes"
 }
 Here "record_no" and "bpname" are mandatory input parameter.
 The "lineitem" is by default "yes", if not provided.
```

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

> **Note**: If Integration interface is defined for that BP. All fields defined in this interface will be sent in the response.

If Integration interface is not defined, then all Custom defined Fields will be sent in response.

Get BP Sample Response


```
{
  "data":
  [
    {
      "uuu_user_id":"a@abc.com",
      "uuu_record_last_update_date":"2018/04/23 11:06",
      "uveFaxTB16":null,
      "uveCertificateNoTB64":null,
      "uvePrimaryContactTB64":"a",
      "title":"v-00101",
      "uveLicenseNoTB16":null,
      "uuu_contact_company":"v-00101",
      "uveCOIAmoutCA":0.0,
      "_bp_lineitems":
      [
        {
          "uirCntctFstNmTB":"a",
          "uuu_user_workphone":null,
          "uuu_tab_id":"List of Contacts",
          "title":"t",
          "ugenAddress1TXT120":null,
          "ugenAddress2TXT120":null,
          "ugenCountryPD":null,
          "ugenCityTXT50":null,
          "short_desc":"Vendor Contact",
          "li_num":1,
```

```
      "uriCntctLstNmTB":"b",
      "ugenStatePD":null,
      "ugenZipCodeTXT16":null,
      "uveEmailTB120":null,
      "ugenAddress3TXT120":null
    }
  ],
  "uvePolicyNoTB32":null,
  "record_no":"VEN-0023",
  "uveVendorTypePD":"Architect",
  "ugenCityTXT50":null,
  "uveCoiExpDOP":null,
  "uuu_dm_publish_path":"v_path",
  "uuu_contact_last_name":"b",
  "ugenZipCodeTXT16":null,
  "uveEmailTB120":"a@abc.com",
  "ugenAddress3TXT120":null,
  "uveReferenceIdTB16":null,
  "uuu_creation_date":"2018/04/12 09:01",
  "ugenRemarksTB4000":null,
  "uvePhoneTB64":null,
  "uveTaxIDTB16":null,
  "uveMinorityBusCB":0,
  "uveInsuranceCoTB32":null,
  "ugenExpirationDateDOP":null,
  "uveWomanOwnedCB":0,
  "uveVendorNameTB50":"v-00101",
  "ugenAddress1TXT120":null,
  "uveDisadvantagedBusCB":0,
  "ugenDiscipline":null,
  "creator_id":"Company Administrator",
  "ugenAddress2TXT120":null,
  "ugenCountryPD":null,
  "uveVendorIDTB16":"v-00101",
```

```
      "ugenStatePD":null,
      "uuu_contact_first_name":"a",
      "status":"Active"
    }
  ],
  "message":
  [ "success"
  ],
  "status":200
}
```

The following funding information for general spends, payment application, or generic cost BPs will be present in the response:

- unassigned_amount
- fund (JSON array containing below)
  - code
  - fund_balance
  - assigned_amount

Get BP Record With Attachments

GET /ws/rest/service/v1/bp/record/file/{project_number}

Purpose:

Get a record detail and all the attachments in the record and its lineitem and general comments, based on input parameter, in of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_num: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

GET URL Parameter

```
input =
 {
     "bpname":"Vendors",
     "record_no":"VEN-0024",
```

```
    "lineitem":"yes",
    "lineitem_file":"yes",
    "general_comments":"yes"
   "attach_all_publications":"yes"
}
```

The "record_no" and "bpname" are mandatory input parameter.

The "lineitem" values "yes" or "no" default "yes", if not provided.

The "lineitem_file" values "yes" or "no", default is "yes", but if "lineitem" is "no" then "lineitem_file" will be considered "no".

The "lineitem_file" input parameter is decide if lineitem attachments will be download or not.

The "general_comments" is for having general comment data in response along with its attachment, by default its "yes".

The "attach_all_publications" is for downloading all the publications/revisions for file attached. by default it "no".

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 otherwise it will be "success".

Record data will be part of response field "record_data", and record level attachments will always return in response.

Attachments will be part of response field "file_handler", which is base64 encoded zip file String and should be decoded with base64 decoder application or online decoder, for example: `https://www.base64decode.org/`

The file name is outer most "file_name" field in response.

If multiple revision or publication of file is attached, then attachment detail in response will contain all the publications; however, the file with latest publication will be part of downloaded attachment.

To Download all the publication/revisions of file attached in record set "attach_all_publications" in input json. It will get all publications from Record upper and detail forms only.

Output Folder structure:

```
zip_file
        |
            +-- attachments
            |           |
            |           +-- files attached to record ( upper form )
            |           |
            |           +-- lineitem_<lineitem number>
```

```
               |                                                |
               |                                                +-- files
attached to lineitem
               |
               +-- gc_attachments ( general comment attachments )
               |
               +-- comment_<comment number>
                                    |
                                    +-- files attached to comment
```

Output Folder structure when "attach_all_publications" is "yes":

```
zip_file
      |
        +-- attachments
        |            |
        |            +-- publication_<publication number>
        |            |             |
        |            |            +-- files attached to record which has
same publication number as in parent folder suffix
        |            |
        |            +-- lineitem_<lineitem number>
        |                         |
        |                        +-- publication_<publication number>
        |                                                  |
        |                                              +-- files attached to
lineitem which has same publication number as in parent folder suffix
        |
       +-- gc_attachments ( general comment attachments )
                   |
                   +-- comment_<comment number>
                                    |
                                    +-- files attached to comment
```

> **Note**: If Integration interface is defined for that BP. All fields defined in this interface will be sent in the response.

If the integration interface is not defined, then all Custom defined Fields will be sent in response.

Get BP with attachment Sample Response


```
{
   "data":[
      {
         "record_data":{
            "uuu_user_id":"a@abc.com",
            "uuu_record_last_update_date":"2018/05/28 04:34",
            "uveFaxTB16":null,
            "uveCertificateNoTB64":null,
            "uvePrimaryContactTB64":"a",
            "title":"v-00024",
            "uveLicenseNoTB16":null,
            "_general_comments":[
               {
                  "firstname":"Company",
                  "comment_state":"Active",
                  "create_date_iso":"2018-05-28T04:34",
                  "page_num":"0",
                  "lastmodified_iso":"2018-05-28T04:34",
                  "shortname":"idc",
                  "content":"\nNew attach comment",
                  "has_annotation":"0",
                  "lastname":"Administrator",
                  "attachment":{
                     "issue_date":null,
                     "revision_no":null,
                     "file_name":"27362157.jpg",
                     "publication_no":1,
                     "title":null,
                     "file_size":201571
                  },
                  "companyname":"idc",
                  "attachment_id":"0",
```

```
            "fullname":"Company Administrator",
            "create_date":[
               "05-28-2018 04:34:55",
               "05-28-2018 04:34:55"
            ],
            "attachment_count":"1"
         },
         {
            "firstname":"Company",
            "comment_state":"Active",
            "create_date_iso":"2018-05-28T04:07",
            "page_num":"0",
            "lastmodified_iso":"2018-05-28T04:07",
            "shortname":"idc",
            "content":"\nThis is a wonderful general comment",
            "has_annotation":"0",
            "lastname":"Administrator",
            "companyname":"idc",
            "attachment_id":"0",
            "fullname":"Company Administrator",
            "create_date":[
               "05-28-2018 04:07:03",
               "05-28-2018 04:07:03"
            ],
            "attachment_count":"0"
         }
      ],
      "uuu_contact_company":"v-00024",
      "uveCOIAmoutCA":0.0,
      "_bp_lineitems":[
         {
            "uirCntctFstNmTB":"a",
            "uuu_user_workphone":null,
            "uuu_tab_id":"List of Contacts",
```

```
      "title":"t",
      "ugenAddress1TXT120":null,
      "attachment":[
         {
            "reference":"NO",
            "issue_date":null,
            "revision_no":null,
            "file_name":"sample_upload_1.pdf",
            "publication_no":1,
            "title":null,
            "create_date":"2018/05/21 19:39",
            "file_size":83768
         }
      ],
      "ugenAddress2TXT120":null,
      "ugenCountryPD":null,
      "ugenCityTXT50":null,
      "short_desc":"Vendor Contact",
      "li_num":1,
      "uriCntctLstNmTB":"b",
      "ugenStatePD":null,
      "ugenZipCodeTXT16":null,
      "uveEmailTB120":null,
      "ugenAddress3TXT120":null
   },
   {
      "uirCntctFstNmTB":"a",
      "uuu_user_workphone":null,
      "uuu_tab_id":"List of Contacts",
      "title":"t",
      "ugenAddress1TXT120":null,
      "attachment":[
         {
            "reference":"NO",
```

```
                "issue_date":null,
                "revision_no":null,
                "file_name":"sample_upload_1.pdf",
                "publication_no":0,
                "title":null,
                "create_date":"2018/05/21 19:39",
                "file_size":83768
            },
            {
                "reference":"NO",
                "issue_date":null,
                "revision_no":null,
                "file_name":"sample_upload_2.pdf",
                "publication_no":1,
                "title":null,
                "create_date":"2018/05/21 19:39",
                "file_size":83768
            }
        ],
        "ugenAddress2TXT120":null,
        "ugenCountryPD":null,
        "ugenCityTXT50":null,
        "short_desc":"Vendor Contact",
        "li_num":2,
        "uriCntctLstNmTB":"b",
        "ugenStatePD":null,
        "ugenZipCodeTXT16":null,
        "uveEmailTB120":null,
        "ugenAddress3TXT120":null
    }
],
"attachment":[
    {
        "reference":"NO",
```

```
            "issue_date":null,
            "revision_no":null,
            "file_name":"sample_upload_3.pdf",
            "publication_no":1,
            "title":null,
            "create_date":"2018/05/21 19:39",
            "file_size":83768
        },
        {
            "reference":"NO",
            "issue_date":null,
            "revision_no":null,
            "file_name":"sample_upload_3.pdf",
            "publication_no":2,
            "title":null,
            "create_date":"2018/05/27 23:55",
            "file_size":83768
        }
    ],
    "uvePolicyNoTB32":null,
    "record_no":"VEN-0024",
    "uveVendorTypePD":"Architect",
    "ugenCityTXT50":null,
    "uveCoiExpDOP":null,
    "uuu_dm_publish_path":"v_path",
    "uuu_contact_last_name":"b",
    "ugenZipCodeTXT16":null,
    "uveEmailTB120":"a@abc.com",
    "ugenAddress3TXT120":null,
    "uveReferenceIdTB16":null,
    "uuu_creation_date":"2018/04/19 02:13",
    "ugenRemarksTB4000":null,
    "uvePhoneTB64":null,
    "uveTaxIDTB16":null,
```

```
            "uveMinorityBusCB":0,
            "uveInsuranceCoTB32":null,
            "ugenExpirationDateDOP":null,
            "uveWomanOwnedCB":0,
            "uveVendorNameTB50":"v-00024",
            "ugenAddress1TXT120":null,
            "uveDisadvantagedBusCB":0,
            "ugenDiscipline":null,
            "creator_id":"Company Administrator",
            "ugenAddress2TXT120":null,
            "ugenCountryPD":null,
            "uveVendorIDTB16":"v-00024",
            "ugenStatePD":null,
            "uuu_contact_first_name":"a",
            "status":"Active"
         },
         "file_name":"_uv_VEN-0024_0_attachments.zip",
         "file_handler":" base64 encoded zip file "
      }
   ],
   "message":[
      "success"
   ],
   "status":200
}
```


The following funding information for general spends, payment application, or generic cost BPs will be present in the response:

- unassigned_amount
- fund (JSON array containing below)
  - code
  - fund_balance
  - assigned_amount

Create BP Record

POST /ws/rest/service/v1/bp/record/{project_number}

Purpose:

Create a record in a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.

```
{
    "options": {
        "bpname": "Vendors",
        "workflow_details":
        {
            "workflow_name": "workflow_name",
            "user_name" : "first_name  last_name",
            "action_name" : "action_name"
        }
    },
    "data": [
            {
                "record_no":"hello",
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
```

```
"_bp_lineitems": [
    {
        "uirCntctFstNmTB": "a",
        "uuu_user_workphone": null,
        "uuu_tab_id": "List of Contacts",
        "title": "t",
        "ugenAddress1TXT120": null,
        "ugenAddress2TXT120": null,
        "ugenCountryPD": null,
        "ugenCityTXT50": null,
        "short_desc": "Vendor Contact",
        "uriCntctLstNmTB": "b",
        "ugenStatePD": null,
        "ugenZipCodeTXT16": null,
        "uveEmailTB120": "hello@abc.com",
        "ugenAddress3TXT120": null
    }
],
"uvePolicyNoTB32": null,
"uveVendorTypePD": "Architect",
"ugenCityTXT50": null,
"uveCoiExpDOP": null,
"uuu_dm_publish_path": "v_path",
"uuu_contact_last_name": "b",
"ugenZipCodeTXT16": null,
"uveEmailTB120": "a@abc.com",
"ugenAddress3TXT120": null,
"uveReferenceIdTB16": null,
"uuu_creation_date": "04-12-2018",
"ugenRemarksTB4000": null,
"uvePhoneTB64": null,
"uveTaxIDTB16": null,
"uveMinorityBusCB": 0,
"uveInsuranceCoTB32": null,
```

```
                "ugenExpirationDateDOP": null,
                "uveWomanOwnedCB": 0,
                "uveVendorNameTB50": "v-00101",
                "ugenAddress1TXT120": null,
                "uveDisadvantagedBusCB": 0,
                "ugenDiscipline": null,
                "creator_id": "Company Administrator",
                "ugenAddress2TXT120": null,
                "ugenCountryPD": null,
                "uveVendorIDTB16": "v-00101",
                "ugenStatePD": null,
                "uuu_contact_first_name": "a",
                "status": "Active"
            }
        ]
}
```


If "record_no" is not provided then auto-generated record_no will be assigned to record, If unique "record_no" is provided then record will be created be given record_no.

- "bpname" is mandatory and is part of options form parameter.
- If "workflow_details" is not specified, then the fall back will be the auto-creation settings defined on the BP Setup.
- If specified, then all the three params - user_name, workflow_name and action_name are mandatory.
- Validations Performed:
  - If the user is a valid active user in the project.
  - Workflow name is valid and active.
  - User is an assignee (user/group) on the creation step of the workflow.
  - Action name is valid outgoing link from the creation step.
  - Error messages:
    - Enter a valid or an active user. (Missing key or invalid user name)
    - The workflow name is not valid. (Missing key or invalid/inactive workflow name)
    - The user does not have an active workflow template. (User is not an assignee on the creation step)
    - The workflow action name is not valid. (Missing key or invalid outgoing action name from creation step)

Output:

JSON object containing 'status', 'data', 'message.'

A message will be present if status is not 200.

Status codes are:

1> 200 OK , if all creation succeeds .

2> 3000 , If any of records creation fail.

Create BP Sample Response


```
{
    "data": [],
    "message": [
        {
                "_record_status": "success",
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
                "_bp_lineitems": [
                    {
                        "uirCntctFstNmTB": "a",
                        "uuu_user_workphone": null,
                        "uuu_tab_id": "List of Contacts",
                        "title": "t",
                        "ugenAddress1TXT120": null,
                        "ugenAddress2TXT120": null,
                        "ugenCountryPD": null,
                        "ugenCityTXT50": null,
                        "short_desc": "Vendor Contact",
                        "uriCntctLstNmTB": "b",
                        "ugenStatePD": null,
                        "ugenZipCodeTXT16": null,
```

```
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null
        }
    ],
    "uvePolicyNoTB32": null,
    "record_no": "hello",
    "_attachment": [
        {
            "file_name": "new 1.txt"
        }
    ],
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
    "ugenZipCodeTXT16": null,
    "uveEmailTB120": "a@abc.com",
    "ugenAddress3TXT120": null,
    "uveReferenceIdTB16": null,
    "uuu_creation_date": "04-12-2018",
    "ugenRemarksTB4000": null,
    "uvePhoneTB64": null,
    "uveTaxIDTB16": null,
    "uveMinorityBusCB": 0,
    "uveInsuranceCoTB32": null,
    "ugenExpirationDateDOP": null,
    "uveWomanOwnedCB": 0,
    "uveVendorNameTB50": "v-00101",
    "ugenAddress1TXT120": null,
    "uveDisadvantagedBusCB": 0,
    "ugenDiscipline": null,
    "creator_id": "Company Administrator",
    "ugenAddress2TXT120": null,
```

```
                "ugenCountryPD": null,
                "uveVendorIDTB16": "v-00101",
                "ugenStatePD": null,
                "uuu_contact_first_name": "a",
                "status": "Active"
            }
    ],
    "status": 200
}
in case of any error :
{
    "data": [],
    "message": [
        {
            "_record_status": "Error Business Process record_no hello
already exists. ",
            "record": {
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
                "_bp_lineitems": [
                    {
                        "uirCntctFstNmTB": "a",
                        "uuu_user_workphone": null,
                        "uuu_tab_id": "List of Contacts",
                        "title": "t",
                        "ugenAddress1TXT120": null,
                        "ugenAddress2TXT120": null,
```

```
            "ugenCountryPD": null,
            "ugenCityTXT50": null,
            "short_desc": "Vendor Contact",
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null
        }
    ],
    "uvePolicyNoTB32": null,
    "record_no": "hello",
    "_attachment": [
        {
            "file_name": "new 1.txt"
        }
    ],
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
    "ugenZipCodeTXT16": null,
    "uveEmailTB120": "a@abc.com",
    "ugenAddress3TXT120": null,
    "uveReferenceIdTB16": null,
    "uuu_creation_date": "04-12-2018",
    "ugenRemarksTB4000": null,
    "uvePhoneTB64": null,
    "uveTaxIDTB16": null,
    "uveMinorityBusCB": 0,
    "uveInsuranceCoTB32": null,
    "ugenExpirationDateDOP": null,
    "uveWomanOwnedCB": 0,
```

```
                "uveVendorNameTB50": "v-00101",
                "ugenAddress1TXT120": null,
                "uveDisadvantagedBusCB": 0,
                "ugenDiscipline": null,
                "creator_id": "Company Administrator",
                "ugenAddress2TXT120": null,
                "ugenCountryPD": null,
                "uveVendorIDTB16": "v-00101",
                "ugenStatePD": null,
                "uuu_contact_first_name": "a",
                "status": "Active"
            }
        }
    ],
    "status": 3000
}
```


Update BP Record

PUT /ws/rest/service/v1/bp/record/{project_number}

Purpose:

Update record in a specific BP in a shell based on shell number or from Company level if project/shell number is not provided. The input JSON provides various options to be considered for fetching the data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be from Company Level.

PUT

```
{
        "options":{
            "bpname":"line6",
            "LineItemIdentifier":"upwoLISAS",
            "workflow_details" :
            {
               "WFCurrentStepName":"Step 4",
```

```
                "WFActionName":"Line 3",
            }
        },
        "data": [
            {
                "ugenUserIDPK":null,
                "_bp_lineitems":
                [
                    {
                    "ugenShellPK":"/AP1",
                    "uuu_tab_id":"Standard",
                    "short_desc":"desc3_m11",
                    "upwoLISAS":"0001",
                    "uuu_line_item_status":"Active"
                    }
                ],
                "record_no":"uxli6-0020",
                "title":"new title"
            }
            ]
    }
```


Notes:

- Auto-Creation setting is mandatory in Shell or Company BP Setup in Admin mode.
- The "record_no" is mandatory and is part of data.
- The "bpname" is mandatory and is part of options form parameter.
- The WFCurrentStepName and WFActionName are part of the "workflow_details" and are for Workflow-type BPs. If the values for the WFCurrentStepName and WFActionName are provided, then the record will be updated and sent to the next step of the workflow. If the values for the WFCurrentStepName and WFActionName are not provided, then the record will be updated but will remain in same workflow step.
- The LineItemIdentifier is a part of options form parameter. The LineItemIdentifier value will be the DE name present in the lineItem. The DE name present in the lineItem should be present in the "data" (input record json) with its value.

Steps to add LineItemIdentifier in the BP design:

1) Create a custom DE with DD: Sys Auto Sequence.
2) Add this DE to the Detail Form of the BP with the line items that need to be updated.
3) Navigate to the Company Workspace > Admin mode > Business Processes and open the BP.
4) Click Open drop-down to open and select Data Elements to open the Data Elements Configuration window.
5) Click the Auto Sequence tab and click Add.

6) Select the data element added in the Detail Form and select the level as: Per Record.
7) Provide the start value and click Create.
8) Add the parameter "Sys Sequence counter" and click OK.
9) Add the DE to Integration > Detail.
10) Set the "Direction" to "Both" for that DE.
11) Complete and deploy the BP.
12) In the body of the request, add the DE name as Lineitem identifier and send the request.

Delete the Line-Item, using Update Request:

To delete the line-Item using update request, the field "_delete_line_item" should be used (value of which will be comma separated values of LineItemIdentifier DE) as shown in the following example:

Sample: Delete line-item input

```
{
                    "options": {
                   "bpname":"line6",
                  "LineItemIdentifier":"upwoLISAS"
                  },
                 "data": [
                {
                    "ugenUserIDPK":null,
                    "_delete_bp_lineitems":"0002,0001,0003",
                    "record_no":"uxli6-0021",
                    "title":"title_m112244"
               }
           ]
      }
```

Note: PUT call has input & output, both as JSON in the body.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 .

Status codes are:

1>200 OK, if all update succeeds .
2>3000, if any of records update fails.

Update BP Sample Response

```
{
    "data": [],
    "message": [
```

```
        {
            "_record_status": "success",
            "record": {
                "ugenUserIDPK": null,
                "record_no": "uxli6-0021",
                "title": "title_m112244"
            }
        }
    ],
    "status": 200
}
```

An Integration user can modify the Fund Code for an existing record line items, by using `UpdateBPRecord` call.

A Unifier user can see the Fund Code for an existing record line items being updated and the line item amount is rolled up to the Fund Sheet, based on updated Fund Code.

Additional Information:

The "record_no" is mandatory, and it is a part of data for the `UpdateBPRecord` call.

The "bpname" is mandatory, and it is a part of options form parameter.

The "WFCurrentStepName" and "WFActionName" are part of the "workflow_details," and they are for Workflow type BPs.

If the values for "WFCurrentStepName" and "WFActionName" are provided, then the record will be updated and sent to the next step of workflow.

If the values for "WFCurrentStepName" and "WFActionName" are not provided, then the record will be updated but will remain in the same workflow step.

The "LineItemIdentifier" is a part of options form parameter. The "LineItemIdentifier" value will be the DE name that is present in the "lineItem." The DE is present in the "data" (input record json) along with its value.

In the "_bp_lineitems" section, if "row_id" is modified along with other attributes, then update the Fund Code for the line item in the "data" (with the "Lineitemidentifier" specified).

Any auto-populate DEs that are based on the "row_id" (Fund Code) will be populated by means of the update.

Formulas based on auto-populate DES will be evaluated in the line item.

When a record is routed to next step for workflow BPs, if the "work_flow" details are specified, then verify that in the Fund sheet the line item amount is rolled up to the modified Fund Code.

The rules that are based on the Fund Code will be evaluated when routing the record through the "UpdateBPRecord" call.

For non-workflow Fund BP, if in the input request the status is specified along with the Fund Code in the line item, upon successful request the existing line item will be updated with the specified Fund Code, and the Fund Sheet will show the line item amount against the modified Fund Code.

> **Note**: For the line item with both Fund Code and CBS Code BPs, only the Fund Code for the existing line items can be modified, by using the "UpdateBPRecord" call.

Data Elements in Line Items

If the following Data Elements are present in the Line Items, they will not be updated during UpdateRecordV2 call.

- `bitemid`
- `budgetid`
- `sovitemid`
- `uuu_cm0_picker`
- `xid`
- `ds_code`
- `fsm_lineitem_id`
- `asm_lineitem_id`
- `uuu_lat`
- `uuu_long`
- `uuu_company_account_name`
- `uuu_company_account_code`
- `uuu_company_acc_codepicker`
- `uuu_activity_picker`
- `uuu_asset_picker`
- `ref_bpo_lineitem`
- `ref_bpo`
- `ref_rfb`
- `refid`
- `uuu_sovlinum`
- `otherCompanyID`
- `uuu_costcode_picker`
- `scheduled_value`
- `uuu_asset_cum_depreciation`
- `uuu_asset_curr_period_dep`
- `uuu_asset_net_book_value`
- `uuu_unit_price`
- `currencyid`
- `due_date`
- `uuu_asset_calc_as_of_date`
- `uuu_from_date`

- uuu_issue_date
- uuu_last_update_date
- uuu_rfb_due_date
- uuu_to_date
- uuu_week_picker
- uuu_datasource
- uuu_depreciation_mtd
- uuu_line_item_status
- row_id
- uuu_bid_count
- uuu_bidders_count
- uuu_planning_item_picker
- uuu_proj_picker
- record_no
- uuu_resc_picker
- uuu_role_picker
- uuu_commit_short_desc
- end_date
- uuu_creation_date
- uuu_asset_name
- uuu_commit_breakdown
- uuu_asset_code
- uuu_asset_navigation_code
- uuu_timescale_units
- creator_id
- wpid
- uuu_cost_costattribute
- uuu_cost_external_refid
- uuu_cost_status
- uuu_cost_cost_type
- uuu_cost_description
- uuu_cost_item
- uuu_cost_owner
- uuu_cost_code
- uuu_dm_create_date
- uuu_file_create_date
- uuu_file_issue_date
- uuu_dm_percent_complete
- uuu_file_size
- uuu_file_version
- uuu_dm_description

- uuu_dm_node_path
- uuu_dm_node_name
- uuu_file_revision_no
- uuu_file_title
- uuu_dm_create_by
- uuu_file_create_by
- uuu_fund_fundcategory
- uuu_fund_long_desc
- uuu_fund_description
- uuu_fund_fundname
- uuu_fund_code
- uuu_description
- uuu_resc_nwd_type
- uuu_resc_interest
- uuu_resc_proficiency
- uuu_resc_skill
- uuu_role_status
- uuu_role_name
- uuu_resc_capacity
- uuu_user_address
- uuu_user_city
- uuu_user_country
- uuu_user_email
- uuu_user_fax
- uuu_user_firstname
- uuu_user_homephone
- uuu_user_lastname
- uuu_user_mobilephone
- uuu_user_pager
- uuu_user_state
- uuu_user_timezone
- uuu_user_title
- uuu_user_workphone
- uuu_user_zip
- uuu_resc_status
- uuu_resc_code
- uuu_resc_name
- uuu_early_finish
- uuu_early_start,uuu_finish
- uuu_late_finish
- uuu_late_start

- uuu_start
- uuu_act_pct_complete
- uuu_activity_work_hrs
- uuu_duration
- uuu_fixed_cost
- uuu_float
- uuu_labor_cost
- uuu_non_labor_cost
- uuu_total_cost
- uuu_activity_id
- uuu_flag_complete
- uuu_flag_milestone
- uuu_activity_resources
- uuu_dependency_type
- uuu_activity_name
- uuu_outline_code
- uuu_cost_li_type

Update BP Record with Attachment (REST API Details in Business Processes)

PUT /ws/rest/service/v1/bp/record/file/{project_number}

Purpose:

Update a record with attachment in a specific BP in a shell based on shell number or from Company level if the project, or shell number, is not provided.

The input JSON shall provide various options to be considered for fetching the data.

Input:

All parameters must be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be from Company Level.

Both input & output in JSON format in the body.

Update BP Record with Attachment

```
{
   "options":{
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS",
      "workflow_details":{
         "WFCurrentStepName":"Step 4",
```

```
        "WFActionName":"Line 3"
      }
   },
   "data":[
      {
         "ugenUserIDPK":null,
         "_bp_lineitems":[
            {
               "ugenShellPK":"/AP1",
               "uuu_tab_id":"Standard",
               "short_desc":"desc3_m11",
               "upwoLISAS":"0001",
               "uuu_line_item_status":"Active"
            }
         ],
         "record_no":"uxli6-0020",
         "title":"new title",
         "_attachment":[
            {
               "file_name":"file_1.txt",
               "title":"file_title1",
               "issue_date":"05/06/2018",
               "revision_no":"300"
            },
            {
               "file_name":"file_2.txt",
               "title":"file_title2",
               "issue_date":"05/06/2018",
               "revision_no":"200"
            }
         ]
      }
   ],
   "_attachment":
```

```
   {
        "zipped_file_name" : "zip_file_name.zip",
        "zipped_file_size": "746089",
        "zipped_file_content" :"<base64_encoded string of
zip_file_name.zip file>"
    }
}
```

Notes:

- If "adding attachment" is disabled in Upper form or Detail form, and user is trying to attach file to corresponding form. Updating record will fail.
- Only one record update with attachments is supported in this call. If the file already exists in record, then that file will be ignored and will not get revised .
- If integration form has "add attachment" disabled, then files will not get attached.
- If the "_attachment" (outside data list) for zip file is not provided, then the record will be updated without attachment.
- Auto-Creation setting is mandatory in Shell or Company BP Setup in the Admin mode.
- In input JSON options, the "project_number" specifies the shell in which the records exist, if the "project_number" is not provided, then the records are considered to be from Company Level.
- The "record_no" is mandatory and is a part of data.
- The "bpname" is mandatory and is a part of options form parameter.
- The WFCurrentStepName and WFActionName are part of the "workflow_details" and are for Workflow type BPs. If the values for WFCurrentStepName and WFActionName are provided, then the record will be updated and sent to next step of the workflow. If the values for WFCurrentStepName and WFActionName are not provided, then the record will be updated but it will remain on the same workflow step.
- The LineItemIdentifier is part of options form parameter. The LineItemIdentifier value will be DE name present in the lineItem. That DE should be present in "data" (input record JSON) with its value.

Steps to add LineItemIdentifier in the BP design:

1. Create a custom DE with DD: Sys Auto Sequence.
2. Add this DE to the Detail Form of the BP with the line items that need to be updated.
3. Navigate to the Company Workspace > Admin mode > Business Processes and open the BP.
4. Click Open drop-down to open and select Data Elements to open the Data Elements Configuration window.
5. Click the Auto Sequence tab and click Add.
6. Select the data element added in the Detail Form and select the level as: Per Record.
7. Provide the start value and click Create.
8. Add the parameter "Sys Sequence counter" and click OK.
9. Add the DE to Integration > Detail.

10. Set the "Direction" to "Both" for that DE.
11. Complete and deploy the BP.
12. In the body of the request, add the DE name as Lineitem identifier and send the request.

Delete the Line-Item, using Update Request:

To delete the line-Item using update request, the field "_delete_line_item" should be used (value of which will be comma separated values of LineItemIdentifier DE) as shown in the following example:

Sample: *Delete line-item input*

```
{
   "options":{
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS"
   },
   "data":[
      {
         "ugenUserIDPK":null,
         "_delete_bp_lineitems":"0002,0001,0003",
         "record_no":"uxli6-0021",
         "title":"title_m112244"
      }
   ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 .

Status codes are:

1>200 OK, if all creation succeeds.
2>3000, if any of records creation fails.

Sample: *Update record with attachment response*

```
{
   "data":[


   ],
   "message":[
```

```
      {
         "_record_status":"success",
         "record":{
            "ugenUserIDPK":null,
            "record_no":"uxli6-0021",
            "title":"title_m112244"
         }
      }
   ],
   "status":200
}
```

or if zip_file_size of uploaded zip file is not correct :

```
   {
       "data": [],
       "message": [
           "Uploaded Zip file is invalid"
       ],
       "status": 3003
   }
```

Data Elements in Line Items

If the following Data Elements are present in the Line Items, they will not be updated during UpdateRecordV2 call.

- bitemid
- budgetid
- sovitemid
- uuu_cm0_picker
- xid
- ds_code
- fsm_lineitem_id
- asm_lineitem_id
- uuu_lat
- uuu_long
- uuu_company_account_name
- uuu_company_account_code
- uuu_company_acc_codepicker
- uuu_activity_picker
- uuu_asset_picker
- ref_bpo_lineitem

- ref_bpo
- ref_rfb
- refid
- uuu_sovlinum
- otherCompanyID
- uuu_costcode_picker
- scheduled_value
- uuu_asset_cum_depreciation
- uuu_asset_curr_period_dep
- uuu_asset_net_book_value
- uuu_unit_price
- currencyid
- due_date
- uuu_asset_calc_as_of_date
- uuu_from_date
- uuu_issue_date
- uuu_last_update_date
- uuu_rfb_due_date
- uuu_to_date
- uuu_week_picker
- uuu_datasource
- uuu_depreciation_mtd
- uuu_line_item_status
- row_id
- uuu_bid_count
- uuu_bidders_count
- uuu_planning_item_picker
- uuu_proj_picker
- record_no
- uuu_resc_picker
- uuu_role_picker
- uuu_commit_short_desc
- end_date
- uuu_creation_date
- uuu_asset_name
- uuu_commit_breakdown
- uuu_asset_code
- uuu_asset_navigation_code
- uuu_timescale_units
- creator_id
- wpid

- uuu_cost_costattribute
- uuu_cost_external_refid
- uuu_cost_status
- uuu_cost_cost_type
- uuu_cost_description
- uuu_cost_item
- uuu_cost_owner
- uuu_cost_code
- uuu_dm_create_date
- uuu_file_create_date
- uuu_file_issue_date
- uuu_dm_percent_complete
- uuu_file_size
- uuu_file_version
- uuu_dm_description
- uuu_dm_node_path
- uuu_dm_node_name
- uuu_file_revision_no
- uuu_file_title
- uuu_dm_create_by
- uuu_file_create_by
- uuu_fund_fundcategory
- uuu_fund_long_desc
- uuu_fund_description
- uuu_fund_fundname
- uuu_fund_code
- uuu_description
- uuu_resc_nwd_type
- uuu_resc_interest
- uuu_resc_proficiency
- uuu_resc_skill
- uuu_role_status
- uuu_role_name
- uuu_resc_capacity
- uuu_user_address
- uuu_user_city
- uuu_user_country
- uuu_user_email
- uuu_user_fax
- uuu_user_firstname
- uuu_user_homephone

- uuu_user_lastname
- uuu_user_mobilephone
- uuu_user_pager
- uuu_user_state
- uuu_user_timezone
- uuu_user_title
- uuu_user_workphone
- uuu_user_zip
- uuu_resc_status
- uuu_resc_code
- uuu_resc_name
- uuu_early_finish
- uuu_early_start,uuu_finish
- uuu_late_finish
- uuu_late_start
- uuu_start
- uuu_act_pct_complete
- uuu_activity_work_hrs
- uuu_duration
- uuu_fixed_cost
- uuu_float
- uuu_labor_cost
- uuu_non_labor_cost
- uuu_total_cost
- uuu_activity_id
- uuu_flag_complete
- uuu_flag_milestone
- uuu_activity_resources
- uuu_dependency_type
- uuu_activity_name
- uuu_outline_code
- uuu_cost_li_type

Create BP Record with Attachment

POST /ws/rest/service/v1/bp/record/file/{project_number}

Purpose:

Create a record in a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.


```
{
    "options": {
        "bpname": "Vendors",
        "workflow_details":
        {
            "workflow_name": "workflow_name",
            "user_name" : "first_name  last_name",
            "action_name" : "action_name"
        }
    },
    "data": [
            {
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
                "_bp_lineitems": [
                    {
                        "uirCntctFstNmTB": "a",
                        "uuu_user_workphone": null,
                        "uuu_tab_id": "List of Contacts",
                        "title": "t",
                        "ugenAddress1TXT120": null,
                        "ugenAddress2TXT120": null,
                        "ugenCountryPD": null,
                        "ugenCityTXT50": null,
                        "short_desc": "Vendor Contact",
```

```
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null,
    "_attachment":[
        {
            "file_name":"new 1.txt",
            "title":"file_title",
            "issue_date":"05/06/2018",
            "revision_no":"100"
        }
        ]
        }
    ],
    "uvePolicyNoTB32": null,
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
    "ugenZipCodeTXT16": null,
    "uveEmailTB120": "a@abc.com",
    "ugenAddress3TXT120": null,
    "uveReferenceIdTB16": null,
    "uuu_creation_date": "04-12-2018",
    "ugenRemarksTB4000": null,
    "uvePhoneTB64": null,
    "uveTaxIDTB16": null,
    "uveMinorityBusCB": 0,
    "uveInsuranceCoTB32": null,
    "ugenExpirationDateDOP": null,
    "uveWomanOwnedCB": 0,
    "uveVendorNameTB50": "v-00101",
    "ugenAddress1TXT120": null,
    "uveDisadvantagedBusCB": 0,
    "ugenDiscipline": null,
    "creator_id": "Company Administrator",
    "ugenAddress2TXT120": null,
    "ugenCountryPD": null,
```

```
                    "uveVendorIDTB16": "v-00101",
                    "ugenStatePD": null,
                    "uuu_contact_first_name": "a",
                    "status": "Active",
                    "_attachment":[
                        {
                            "file_name":"new 1.txt",
                            "title":"file_title1",
                            "issue_date":"05/06/2018",
                            "revision_no":"300"
                        },
                        {
                            "file_name":"exp1.txt",
                            "title":"file_title2",
                            "issue_date":"05/06/2018",
                            "revision_no":"200"
                        }
                        ]
                }
            ]
        ,
        "_attachment": {
                 "zipped_file_name" : "create.zip",
                 "zipped_file_size": "746089",
                 "zipped_file_content" :"< base64_encoded string of
    create.zip file>"
        }
    }
```

If "adding attachment" is disabled in Upper form or Detail form, and user is trying to attach file to corresponding form. Creating record will fail.

If "record_no" is not provided, then the auto-generated record_no will be assigned to the record

If a unique "record_no" is provided, then the record will be created be given record_no.

- "bpname" is mandatory and is part of options form parameter.
- If "workflow_details" is not specified, then the fall back will be the auto-creation settings defined on the BP Setup.
- If specified, then all the three params - user_name, workflow_name and action_name are mandatory.
- Validations Performed:
  - If the user is a valid active user in the project.
  - Workflow name is valid and active.

- User is an assignee (user/group) on the creation step of the workflow.
- Action name is valid outgoing link from the creation step.
- Error messages:
  - Enter a valid or an active user. (Missing key or invalid user name)
  - The workflow name is not valid. (Missing key or invalid/inactive workflow name)
  - The user does not have an active workflow template. (User is not an assignee on the creation step)
  - The workflow action name is not valid. (Missing key or invalid outgoing action name from creation step)
- If "_attachment" (outside data list) for zip file is not provided, then record will be created without attachment.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200.

Status codes are:

1> 200 OK , if all creation succeeds .

2> 3000 , If any of records creation fail.

> **Note**: Only one record input should be provided in Create BP record with attachment.

Create BP Sample Response


```
{
    "data": [],
    "message": [
        {
            "uuu_user_id": "a@abc.com",
            "uuu_record_last_update_date": "04-12-2018",
            "uveFaxTB16": null,
            "uveCertificateNoTB64": null,
            "uvePrimaryContactTB64": "a",
            "title": "v-00101",
            "uveLicenseNoTB16": null,
            "uuu_contact_company": "v-00101",
            "_record_status": "success",
            "uveCOIAmoutCA": 0,
            "_bp_lineitems": [
                {
                    "uirCntctFstNmTB": "a",
                    "uuu_user_workphone": null,
```

```
            "uuu_tab_id": "List of Contacts",
            "title": "t",
            "ugenAddress1TXT120": null,
            "_attachment": [
                {
                    "issue_date": "05/06/2018",
                    "revision_no": "100",
                    "file_name": "new 1.txt",
                    "title": "file_title"
                }
            ],
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "ugenCityTXT50": null,
            "short_desc": "Vendor Contact",
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null
        }
    ],
    "uvePolicyNoTB32": null,
    "_attachment": [
        {
            "issue_date": "05/06/2018",
            "revision_no": "300",
            "file_name": "new 1.txt",
            "title": "file_title1"
        },
        {
            "issue_date": "05/06/2018",
            "revision_no": "200",
            "file_name": "exp1.txt",
            "title": "file_title2"
        }
    ],
    "record_no": "VEN-0086",
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
```

```
            "uveCoiExpDOP": null,
            "uuu_dm_publish_path": "v_path",
            "uuu_contact_last_name": "b",
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "a@abc.com",
            "ugenAddress3TXT120": null,
            "uveReferenceIdTB16": null,
            "uuu_creation_date": "04-12-2018",
            "ugenRemarksTB4000": null,
            "uvePhoneTB64": null,
            "uveTaxIDTB16": null,
            "uveMinorityBusCB": 0,
            "uveInsuranceCoTB32": null,
            "ugenExpirationDateDOP": null,
            "uveWomanOwnedCB": 0,
            "uveVendorNameTB50": "v-00101",
            "ugenAddress1TXT120": null,
            "uveDisadvantagedBusCB": 0,
            "ugenDiscipline": null,
            "creator_id": "Company Administrator",
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "uveVendorIDTB16": "v-00101",
            "ugenStatePD": null,
            "uuu_contact_first_name": "a",
            "status": "Active"
        }
    ],
    "status": 200
}

in case of any error :
{
    "data": [],
    "message": [
        {
            "_record_status": "Error Business Process record_no hello
already exists. ",
            "record": {
                "uuu_user_id": "a@abc.com",
```

```
"uuu_record_last_update_date": "04-12-2018",
"uveFaxTB16": null,
"uveCertificateNoTB64": null,
"uvePrimaryContactTB64": "a",
"title": "v-00101",
"uveLicenseNoTB16": null,
"uuu_contact_company": "v-00101",
"uveCOIAmoutCA": 0,"uveTaxIDTB16": null,
"_bp_lineitems": [
    {
        "uirCntctFstNmTB": "a",
        "uuu_user_workphone": null,
        "uuu_tab_id": "List of Contacts",
        "title": "t",
        "ugenAddress1TXT120": null,
        "ugenAddress2TXT120": null,
        "ugenCountryPD": null,
        "ugenCityTXT50": null,
        "short_desc": "Vendor Contact",
        "uriCntctLstNmTB": "b",
        "ugenStatePD": null,
        "ugenZipCodeTXT16": null,
        "uveEmailTB120": "hello@abc.com",
        "ugenAddress3TXT120": null
    }
],
"uvePolicyNoTB32": null,
"record_no": "hello",
"_attachment": [
    {
        "file_name": "new 1.txt"
    }
],
"uveVendorTypePD": "Architect",
"ugenCityTXT50": null,
"uveCoiExpDOP": null,
"uuu_dm_publish_path": "v_path",
"uuu_contact_last_name": "b",
"ugenZipCodeTXT16": null,
"uveEmailTB120": "a@abc.com",
```

```
                    "ugenAddress3TXT120": null,
                    "uveReferenceIdTB16": null,
                    "uuu_creation_date": "04-12-2018",
                    "ugenRemarksTB4000": null,
                    "uvePhoneTB64": null,
                    "uveTaxIDTB16": null,
                    "uveMinorityBusCB": 0,
                    "uveInsuranceCoTB32": null,
                    "ugenExpirationDateDOP": null,
                    "uveWomanOwnedCB": 0,
                    "uveVendorNameTB50": "v-00101",
                    "ugenAddress1TXT120": null,
                    "uveDisadvantagedBusCB": 0,
                    "ugenDiscipline": null,
                    "creator_id": "Company Administrator",
                    "ugenAddress2TXT120": null,
                    "ugenCountryPD": null,
                    "uveVendorIDTB16": "v-00101",
                    "ugenStatePD": null,
                    "uuu_contact_first_name": "a",
                    "status": "Active"
                }
            }
        ],
        "status": 3000
    }
    or if zip_file_size of uploaded zip file is not correct :
    {
        "data": [],
        "message": [
            "Uploaded Zip file is invalid"
        ],
        "status": 3003
    }
```


Fetch BP Record List with filter_criteria

POST /ws/rest/service/v1/bp/records/{project_number}

Purpose:

Earlier in REST V1 service, "POST /ws/rest/service/v1/bp/records/{projectnumber}" used "filter_condition" to filter the BP records. Only '=' (equal) condition was supported for Data Elements 'end_date','record_no' and 'status'"

filter_condition

```
{
    "bpname" : "Vendors",
    "lineitem" : "yes",
    "lineitem_fields" : " uuu_tab_id;title",
    "filter_condition" : "status=Active",
    "record_fields" : "uuu_user_id;uuu_record_last_update_date"
}
```

Enhancing BP Record list V1 REST API to support for filters like:

- Text, String = Pattern matching
- Number, Cost = Ranges and arithmetic / logical operators
- Date = Ranges

To support the new BP filters, new "**filter_criteria**" should be added in POST data.

filter_criteria

```
{
    "bpname" : "Vendors",
    "lineitem" : "yes",
    "lineitem_fields" : "uuu_tab_id;title",
    "record_fields" : "uuu_user_id;uuu_record_last_update_date",
    "filter_criteria":{
        "join":"OR",
        "filter":[
            {
                "field":"record_no",
                "value":"VEN",
                "condition_type":"like"
            },
            {
                "field":"uuu_file_title",
                "value":"first document title",
                "condition_type":"like"
```

```
            },
            {
                "field":"uuu_file_issue_date",
                "value":"12-07-2017 07:54:00",
                "value2":"20-07-2018",
                "condition_type":"range"
            }
        ]
    }
}
```

filter_criteria key Definitions:

1) The "join": (optional) Specify if the filter condition has to be combined using OR or AND operator. Default is AND.
2) The "filter": List of filter conditions. Each condition will have "field", "value" ("value2" if condition_type is range) and condition_type.
   - The "field": BP form DE name for which the filter condition has to be applied.
   - The "value": value for the DE to filter the data.
   - The "value2": second value for the range condition. Not required to be specified for other conditions.
   - The "condition_type": the filter condition that needs to be applied.

Possible value for condition_type:

condition_type	Notes	Supported Data Element Types
like	For string data types. The value can contain the SQL wildcard characters when like is specified.	Text (String)
eq	For string, numeric and date data type. Will perform the exact match.	Text (String), Number, Cost, Date
lt	For numeric values. Will perform less than value.	Number, Cost, Date
gt	For numeric values. Will perform greater than value.	Number, Cost, Date
lteq	For numeric values. Will perform less than or equal value.	Number, Cost, Date
gteq	For numeric values. Will perform greater than or equal value.	Number, Cost, Date

neq	For numeric values. Will perform not equal value.	Text (String), Number, Cost, Date
range	Numeric, Date values. Will perform range search between value (from) and value2 (to).	Number, Cost, Date

> **Note**: The **Input Date** fields should be provided in "MM-dd-yyyy HH:mm:ss" or ""MM-dd-yyyy" format in **filter_criteria**.

Additional notes:

- All parameters should be URL encoded.
- POST body is a JSON, POST call has input & output both as JSON in the body "bpname" is mandatory.
- The "record_fields" (optional) is list of fields. That is to state that the upper form DR names in the record upper form, only those fields (along with the "record_no) will be listed in the result.
- The "lineitem_fields" (optional) is a list of fields in the record lineitem form, and only those fields will be listed in the result.
- Earlier "filter_condition" is still supported, and it is the condition on the record level, the records which satisfy that criteria will be fetched. Only "=" (equal) condition is supported.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 otherwise it will be "success".

Sample Response of Get BP list after applying filter_criteria


```
{
  "data":
  [
    {
      "uuu_user_id":"a@abc.com",
      "uuu_record_last_update_date":"2018/04/23 11:06",
      "_bp_lineitems":
      [
        {
          "uuu_tab_id":"List of Contacts",
          "li_num":1,
          "title":"t"
        }
      ],
      "record_no":"VEN-0023"
```

```
    },
    {
      "uuu_user_id":"a@abc.com",
      "uuu_record_last_update_date":"2018/05/28 04:34",
      "_bp_lineitems":
      [
        {
          "uuu_tab_id":"List of Contacts",
          "li_num":1,
          "title":"t"
        },
        {
          "uuu_tab_id":"List of Contacts",
          "li_num":2,
          "title":"t"
        }
      ],
      "record_no":"VEN-0024"
    }
  ],
  "message":
  [ "success"
  ],
  "status":200
}
```

REST API Details in Shell Manager

- Create Shell
- Update Shell
- Get Shell
- Get BPs in Shell or Company
- Get Project Shell List
- Create Project

In This Section

Create Shell....267
Update Shell....268
Get Shell....270
Get BPs in Shell or Company....272
Get Project Shell List....273
Create Project....274

Create Shell

POST /ws/rest/service/v1/admin/shell

Input JSON:

```
{
"options" : {
"copy_from_shell_template" : "OFF-00",
"copy_modules" : "",
"default_currency" : ""
},
"data" :
[
{
"uBuildingNumber" : "BLD - 31",
"uBuildingName" : "BUILDING 31",
"uuu_location" : "/All Properties",
"uuu_administrator" : "Company Administrator"
},
{
"uBuildingNumber" : "BLD - 32",
"uBuildingName" : "BUILDING 32",
"uuu_location" : "/All Properties",
"uuu_administrator" : "Company Administrator"
```

```
   }
   ]
}
```

Output JSON:

```
   {
   "data":
   [
   ],
   "message":
   [
   {
   "uBuildingNumber":"BLD - 31",
   "uuu_administrator":"Eadlin Jay",
   "_record_status":"uBuildingName BUILDING 31 already exists under /All
   Properties. ",
   "uuu_location":"/All Properties",
   "uBuildingName":"BUILDING 31"
   },
   {
   "uBuildingNumber":"BLD0032",
   "uuu_administrator":"Eadlin Jay",
   "_record_status":"SUCCESS",
   "uuu_location":"/All Properties",
   "uBuildingName":"BUILDING 32"
   }
   ],
   "status":3000
   }
```

- If all the shells are created successfully, then the response status code will be 200.
- The copy_modules and default_currency are optional.
- The response will have the input data sent in plus the _record_status.
- The _record_status will indicate if the shell creation is a success or the respective error message.

> **Note**: The shell number in the response might not be the same as the input if auto-numbering is set.

Update Shell

PUT /ws/rest/service/v1/admin/shell

Input JSON


```
{
"options" : {
"shelltype" : "All Properties"
},
"data" :
[
{
"uPropertyNumber" : "RE00000",
"description" : "TEST35"
} ,
{
"uPropertyNumber" : "BLD0005",
"description" : "TEST35"
}
]
}
```


Output JSON


```
{
"data":
[
],
"message":
[
{
"_record_status":"SUCCESS",
"description":"TEST35",
"uPropertyNumber":"RE00000"
},
{
"_record_status":"Shell Number is not correct. ",
"description":"TEST35",
"uPropertyNumber":"BLD0005"
}
],
"status":3000
}
```


If all the shells are updated successfully, then the response status code will be 200.

The Response will have the input data sent in plus the _record_status.

The _record_status will indicate if the shell update is a success or the respective error message.

Get Shell

GET /ws/rest/service/v1/admin/shell

Input JSON

Optional filter condition can be provided as parameter to the GET request .

Name of key is options and value is in JSON format as below:

```
{
  "filter": {
                  "shell_type" : "Buildings"
              }
 }
```

Only one value for shell_type is supported.

Output JSON

1) When filter condition not provided, Response will contain all the shells (not template) which are in "Active" status.

```
{
    "data":
    [
        {
            "shell_model_name":"us_apr",
            "shell_number":"RE00000",
            "shell_name":"All Properties",
            "shell_phase":null,
            "latitude":null,
            "longitude":null,
            "shell_location":null
        },
        {
            "shell_model_name":"us_st",
            "shell_number":"site-001",
            "shell_name":"Site_1",
            "shell_phase":null,
            "latitude":null,
            "longitude":null,
            "shell_location":"/All Properties"
        },
        {
```

```
            "shell_model_name":"us_bld",
            "shell_number":"B-0001",
            "shell_name":"Building_1",
            "shell_phase":null,
            "latitude":null,
            "longitude":null,
            "shell_location":"/All Properties/Site_1"
        }
    ],
    "message":
    [ ""
    ],
    "status":200
}
```


2) When filter condition is provided (shell type "Buildings") , then Response will contain shells only of type "Buildings".


```
{
    "data":
    [
        {
            "shell_model_name":"us_bld",
            "shell_number":"B-0001",
            "shell_name":"Building_1",
            "shell_phase":null,
            "latitude":null,
            "longitude":null,
            "shell_location":"/All Properties/Site_1"
        }
    ],
    "message":
    [ ""
    ],
    "status":200
}
```


3) If incorrect shell_type is provided in filter condition, then Error with status 2051 and message "Invalid Shell Type : input shell type " would be the Response.


```
{
    "data":
    [
```

```
    ],
    "message":
    [ "Invalid Shell Type : Buildings_1"
    ],
    "status":2051
}
```

Get BPs in Shell or Company

GET /ws/rest/service/v1/admin/bps/project_number

This GET service results all the Business Processes available (BP set ups) in shell:

- If project_number in URL is provided, or
- The company Level, if project_number is left empty.

Output JSON

1) When project number is provided, for example: /ws/rest/service/v1/admin/bps/B-0001, the Response will contain all the shell level Business Processes available to be used by users in that shell/project. When project number is not provided, for example: /ws/rest/service/v1/admin/bps, the Response will contain all the company level Business Processes available to be used by users in the Company Level.

```
{
 "data":
    [
        {
            "bp_model_name":"uab",
            "bp_name":"Annual Budget",
            "studio_source":"cost",
            "studio_type":"other",
            "no_workflow":false
        },
        {
            "bp_model_name":"uatc",
            "bp_name":"Assets Creator",
            "studio_source":"simple",
            "studio_type":null,
            "no_workflow":false
        },
        {
            "bp_model_name":"ubcfm",
            "bp_name":"Budget Changes-FM",
            "studio_source":"cost",
```

```
            "studio_type":"other",
            "no_workflow":false
         }
      ],
      "message":
      [ ""
      ],
      "status":200
  }
```

2) When incorrect shell number is provided, the Result would be error response with status 602 and message: "Project/Shell Number is not correct.," as shown below.

```
  {
       "data":
        [
        ],
        "message":
        [ "Project/Shell Number is not correct."
        ],
        "status":602
   }
```

Get Project Shell List

GetProjectShellList

GET /ws/rest/service/v1/admin/projectshell

Purpose:

Get Project and Shells

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Url Parameter

"options" (Optional): options attribute can be used to filter the list with the possible attributes.
"status"(Optional): "Active, Inactive, On-Hold, View-Only" are possible values.
"type"(Optional):"wbs_shell,generic_shell" are possible types.
"filter_condition"(Optional): projectname or projectnumber.

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK if success

For error scenarios, the relevant message will be displayed.

GetProjectShellList


```
{
  "data":
  [
    {
      "projectname":"All Projects",
      "projectnumber":"AP",
      "status":"Active",
      "type":"wbs_shell"
    },
    {
      "projectname":"Fred's Test Project 1",
      "projectnumber":"P-0001",
      "status":"Active",
      "type":"wbs_shell"
    },
    {
      "projectname":"Ray's Test Project 1",
      "projectnumber":"P-0002",
      "status":"Active",
      "type":"wbs_shell"
    }
],
  "message":
  [ "Success"
  ],
  "status":200
}
```


Create Project

POST /ws/rest/service/v1/admin/project

Purpose:

To create project

Input:

Both input & output in JSON format in the body. This works only at company level.

Following are the fields that are supported under 'options' object

Element Name	Required	Description
copy_from_project_template	Yes	Specify project number or project template number
copyModules	No	Option to either copy modules or not. This key can only go as a child element under options object. This key will be used to identify whether to create a new project by copying all modules including user / group or just creates a new project by copying user / group from another project or template. Only valid values are Yes / No. Yes means copy all modules including user / groups. No means copy only user / groups. If this key is not provided or value is empty then default behavior will be to create project by copying user / groups.

Following are the fields that are supported under 'data' list object

Element Name	Required	Description
projectnumber	Yes	Project number of the new project that is getting created. Multiple projects with same project number can exist. User defined number.

projectname	Yes	Project name of the new project that is getting created. If this project name already exists you will get an error. User defined name.
projectCurrency	No	Option to specify the project currency. Valid values for this field are the code for the currency rather than the names. For example, USD is valid, United States Dollars is not. If the code is not a valid code then the web service will fail. If this key is not provided or value is empty then projectCurrency value of the template will be taken.
constructiontype	No	Type of construction (New Construction or Retrofit/Remodel).
status	No	Status of the Project (Active, Inactive, On-Hold)
typeofproject	No	Type of Project. Values are based on the user dataset for "Project Type" data definition
projectsite	No	Project site. Values are based on the user dataset for "Project Site" data definition
startdate	No	Start Date of the project.
plannedcompletion	No	Planned Completion date
revisedcompletion	No	Revised Completion date
designcomplete	No	Design % complete
constrcomplete	No	Construction % complete

notes	No	Notes
schedulestatus	No	Schedule status of the project
projectphase	No	Project phase information.Values are based on the user dataset for "Project Phase" data definition
description	No	Description of the project

Following are the fields that are supported under 'location' object

Element Name	**Required**	**Description**
location	No	Base object for project location/address information. If this location key is not provided then Primavera Unifier will create Project by copying "Project Address" information from template.
address	No	Base object for address information. This object can be repeated to provide multiple address types
address1	Yes	Address 1 field of the project. This field is child element of 'address' object. This element is required if address object is provided
address2	No	Address 2 field of the project. This field is child element of 'address' object
city	Yes	City field of the project and it is child element of 'address' object. This element is required if address object is provided

state	Yes	State field of the project and it is child element of 'address' object. This element is required if address object is provided
country	Yes	Country field of the project and it is child element of 'address' object. This element is required if address object is provided
phone	No	Phone field of the project and it is child element of 'address' object.
fax	No	Fax filed of the project and it is child element of 'address' object.
addresstype	Yes	Type of address and it is child element of 'address' object. Following are the valid values for this field ▸ Project Address ▸ Billing ▸ Shipping ▸ Billing and Shipping

Notes:

- Integration users should have 'create' permission of 'Company Administration' permission set to create project through REST services.
- If any non required values are not provided in the input JSON then the respective values from the template will be taken.

Create Project input JSON

```
{
  "options":
  {
    "copy_from_project_template":"ProjectTemplate1",
    "project_currency": "AFN",
  },
  "data": [
```

```
{
    "projectnumber":"Project1",
    "projectname":"Rest Project 1",
    "constructiontype:"New Construction",
    "status":"Active",
    "location":{
        "address":[
            {
                "address1": "Address1",
                "address2": "Address2",
                "addresstype": "Project Address",
                "city": "Hyderabad",
                "state": "Telangana",
                "country": "India",
                "phone": 9999988889,
                "zip": "506454"
            },
            {
                "address1": "Xyz",
                "address2": "Pqr",
                "addresstype": "Billing",
                "city": "Hyderabad",
                "state": "Telangana",
                "country": "India",
                "phone": 9999999999,
                "zip": "516454"
            }
        ]
    }
},
{
    "projectnumber":"Project2",
    "projectname":"Rest Project 2"
}
```

```
  ]
}
```

Output:

JSON object containing '**status**', '**data**', '**message**'

Create Project output JSON

```
{
  "data":
  [
  ],
  "message":
  [
    {
      "projectnumber":"Project 1",
      "projectname":"Rest Project 1",
      "message":"Project Template/Project name already exists. ",
      "status":612
    },
    {
      "projectnumber":"Project 2",
      "projectname":"Rest Project 2",
      "message":"success",
      "status":200
    }
  ],
  "status":3000
}
```

Status codes are:

1> 200 , for success

2> 3000, for partial success

REST API Details in Level

- Authorization
- Response Error Codes
- Get Level List
- Create Level
- Update Level

In This Section

Authorization 281
Response Error Codes 281
Get Level List 282
Create Level 285
Update Level 287

Authorization

Refer to Authentication section in ***REST Web Services V1*** (on page 171)

Response Error Codes

Code	Message	Details
200	success	If the request is successful
500	Server Error, contact the system administrator.	If there is any exception when running this service
602	Project/Shell Number is not correct	project number is incorrect or does not exists
603	Invalid 'space_type' provided in filter	
603	invalid 'space_type' provided in options	
618	Shell Number is not correct.	If incorrect project_number is provided or project_number is not provided in input
695	Project Number cannot be blank	
811	Invalid Filter Condition.	

813	Not supported for Standard project/Programs	
855	Space Type is inactive at company level	
855	Level is inactive at company level	
1101	Empty or Invalid JSON data	If required input JSON is empty
3000	Partial failure.	Partial failure of REST service
3002	Invalid input	If input JSON didn't contain the "data"/required parameter

Get Level List

GET /ws/rest/service/v1/level

Purpose:

Get list of level record from Unifier. This works only at project or shell level.

Input:

All parameters should be URL encoded.

URL query parameter -

- project_number(Required): specify the project/shell number to get project/shell level list.
- filter_condition

filter_condition

```
"filter_condition" : {
    "record_fields" : "uuu_floor_id, uuu_floor_name",
    "filter_criteria":{
        "join":"OR",
        "filter":[
            {
                "field":"status",
                "value":"Vacant",
                "condition_type":"like"
            },
            {
```

```
                "field":"App_date",
                "value":"10-09-2019 07:54:00",
                "value2":"10-17-2019",
                "condition_type":"range"
            }
        ]
    }
}
```

filter_condition key definition

- "record_fields" (optional) is list of fields i.e. DE names in Level , only those fields will be listed out in result.

filter_criteria key Definitions:

1. "join": (optional)Specify if the filter condition has to be combined using OR or AND operator. Default is AND.
2. "filter": List of filter conditions. Each condition will have "field", "value" ("value2" if condition_type is range) and condition_type.
    - "field": BP form DE name for which the filter condition has to be applied.
    - "value": value for the DE to filter the data.
    - "value2": second value for the range condition. Not required to be specified for other conditions.

Possible value for condition_type:

condition_type	notes	Supported Data Element Types
like	For string data types. The value can contain the SQL wildcard characters when like is specified	Text (String)
eq	For string, numeric and date data type. Will perform the exact match.	Text (String), Number, Cost, Date
lt	For numeric values. Will perform less than value.	Number, Cost, Date
gt	For numeric values. Will perform greater than value.	Number, Cost, Date
lteq	For numeric values. Will perform less than or equal value.	Number, Cost, Date

gteq	For numeric values. Will perform greater than or equal value.	Number, Cost, Date
neq	For numeric values. Will perform not equal value.	Text (String), Number, Cost, Date
range	Numeric, Date values. Will perform range search between value and value2	Number, Cost, Date

Note: Input Date fields should be provided in "MM-dd-yyyy HH:mm:ss" or ""MM-dd-yyyy" format in filter_criteria.

Output:

JSON object containing '**status**', '**data**', '**message**'

Get Level List output JSON


```
{
  "data": [
    {
      "App_date": "10/Oct/2019 10:00 AM",
      "uuu_sp_level_name": "F-1008"
    },
    {
      "App_date": "10/Oct/2019 12:00 AM",
      "uuu_sp_level_name": "F-1009"
    },
    {
      "App_date": "10/Oct/2019 12:00 AM",
      "uuu_sp_level_name": "F-1010"
    },
    {
      "App_date": "10/Oct/2019 12:00 AM",
      "uuu_sp_level_name": "F-1011"
    }
  ],
  "message": [],
  "status": 200
}
```

Status codes are:

1> 200 , for success

Create Level

POST /ws/rest/service/v1/level

Purpose:

To create level record. This works only at project or shell level.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create Level input JSON

```
{
 "options" :
  {
      "project_number":"Shell/Project number"
  },
  "data": [
    {
      "uuu_sp_level_name": "F-0001",
      "status": "Occupied",
      "App_date": "25/Dec/2019 7:00 PM",
      "ulevStorAreaND": "0.00000000",
      "ulevGrossBuildAreaNLD": "0.00000000",
      "uuu_sp_uom": "Sq Ft",
      "ulevStorAreaBasNLD": "0.00000000",
      "ulvlCode": "1.00000000",
      "ulevStackOrdDA": "123.00000000",
      "NumericPullDown": "One",
      "multiselect": "One",
      "CPLGrossAreaND": 2345.0
     }
  ]
}
```

Input JSON field description:

- uuu_sp_level_name(Required): specify the floor name
- project_number(Required): specify the project/shell number to create level in that project

Note: Input Date fields should be provided in Integration user preference format in input JSON.

Output:

JSON object containing '**status**', '**data**', '**message**'

Create Level output JSON


```
{
  "data": [
    {
      "uuu_sp_level_name": "F-0001",
      "status": "Occupied",
      "App_date": "25/Dec/2019 7:00 PM",
      "ulevStorAreaND": "0.00000000",
      "ulevGrossBuildAreaNLD": "0.00000000",
      "uuu_sp_uom": "Sq Ft",
      "ulevStorAreaBasNLD": "0.00000000",
      "ulvlCode": "1.00000000",
      "ulevStackOrdDA": "123.00000000",
      "NumericPullDown": "One",
      "multiselect": "One",
      "CPLGrossAreaND": 2345.0
      "ulevMajorVertPenND": 0.0,
      "uuu_sp_level_drawing": null,
      "ulevBuildRatioNLD": 0.0
    }
  ],
  "message": [
    {
      "uuu_sp_level_name": "F-0001",
      "message": "success"
    }
  ],
  "status": 200
}
```

Status codes are:

1> 200 , for success
2>3000, for partial create.

Update Level

PUT /ws/rest/service/v1/level

Purpose:

To update level record. This works only at project or shell level.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update Level input JSON

```
{
  "options" :
  {
      "project_number":"Shell/Project number"
  },
  "data": [
    {
      "uuu_sp_level_name": "F-0001",
      "status": "Occupied",
      "App_date": "26/Dec/2019 1:00 PM",
      "NumericPullDown": "One, Two"
      }
   ]
}
```

Input JSON field description:

- project_number (Required): specify the project/shell number to update level.
- uuu_sp_level_name(Required): specify the floor name to update level

Note: Input Date fields should be provided in Integration user preference format in input JSON.

Output:

JSON object containing '**status**', '**data**', '**message**'

Update Level output JSON

```
{
```

```
  "data": [
    {
      "uuu_sp_level_name": "F-0001",
      "status": "Occupied",
      "App_date": "26/Dec/2019 07:00 PM",
      "ulevStorAreaND": "0.00000000",
      "ulevGrossBuildAreaNLD": "0.00000000",
      "uuu_sp_uom": "Sq Ft",
      "ulevStorAreaBasNLD": "0.00000000",
      "ulvlCode": "1.00000000",
      "ulevStackOrdDA": "123.00000000",
      "NumericPullDown": "One",
      "multiselect": "One, Two",
      "CPLGrossAreaND": 2345.0
      "uuu_sp_level_drawing": null,
      "ulevBuildRatioNLD": 0.0
    }
  ],
  "message": [
    {
      "uuu_sp_level_name": "F-0001",
      "message": "success"
    }
  ],
  "status": 200
}
```


Status codes are:

1> 200 , for success

2> 3000, for partial update.

REST API Details in Space

- Authorization
- Response Error Codes
- Get Space List
- Create Space
- Update Space

In This Section

Authorization 289
Response Error Codes 289
Get Space List 290
Create Space 293
Update Space 295

Authorization

Refer to Authentication section in ***REST Web Services V1*** (on page 171)

Response Error Codes

Code	Message	Details
200	success	If the request is successful
500	Server Error, contact the system administrator.	If there is any exception when running this service
602	Project/Shell Number is not correct	project number is incorrect or does not exists
603	Invalid 'space_type' provided in filter	
603	invalid 'space_type' provided in options	
618	Shell Number is not correct.	If incorrect project_number is provided or project_number is not provided in input
695	Project Number cannot be blank	
811	Invalid Filter Condition.	

813	Not supported for Standard project/Programs	
855	Space Type is inactive at company level	
855	Level is inactive at company level	
1101	Empty or Invalid JSON data	If required input JSON is empty
3000	Partial failure.	Partial failure of REST service
3002	Invalid input	If input JSON didn't contain the "data"/required parameter

Get Space List

GET /ws/rest/service/v1/space

Purpose:

Get list of Space records from Unifier. This works only at project or shell .

Input:

All parameters should be URL encoded.

URL query parameter -

- project_number(Required): specify the project/shell number to get project/shell Space record list.
- filter_condition : Required parameter, as "space_type" (Space bp name) is always required.

URL query parameters

```
project_number = Building_shell_number
filter_condition = {
    "space_type" : "space_bp_name",
    "record_fields":"space_sp_space_name; uuu_rsv_overbook;
uuu_sp_level_picker; uSiteName; uBuildingName",
    "filter_criteria":{
        "join":"AND",
        "filter":[
           {
```

```
            "field":"space_sp_space_name",
            "value":"Space_00%",
            "condition_type":"like"
         },
         {
            "field":"uuu_rsv_overbook",
            "value":"No overbooking",
            "condition_type":"eq"
         }
      ]
   }
}
```

filter_condition key definition

- "space_type" is mandatory .
- "record_fields" (optional) is list of fields i.e. DE names in Space , only those fields (along with space_sp_space_name & uuu_sp_level_picker) will be listed out in result.

filter_criteria key Definitions:

1. "join": (optional)Specify if the filter condition has to be combined using OR or AND operator. Default is AND.
2. "filter": List of filter conditions. Each condition will have "field", "value" ("value2" if condition_type is range) and condition_type.
   - "field": BP form DE name for which the filter condition has to be applied.
   - "value": value for the DE to filter the data.
   - "value2": second value for the range condition. Not required to be specified for other conditions.

Possible value for condition_type:

condition_type	notes	Supported Data Element Types
like	For string data types. The value can contain the SQL wildcard characters when like is specified	Text (String)
eq	For string, numeric and date data type. Will perform the exact match.	Text (String), Number, Cost, Date

lt	For numeric values. Will perform less than value.	Number, Cost, Date
gt	For numeric values. Will perform greater than value.	Number, Cost, Date
lteq	For numeric values. Will perform less than or equal value.	Number, Cost, Date
gteq	For numeric values. Will perform greater than or equal value.	Number, Cost, Date
neq	For numeric values. Will perform not equal value.	Text (String), Number, Cost, Date
range	Numeric, Date values. Will perform range search between value and value2	Number, Cost, Date

Note: Input Date fields should be provided in "MM-dd-yyyy HH:mm:ss" or ""MM-dd-yyyy" format in filter_criteria.

Output:

JSON object containing '**status**', '**data**', '**message**'

Get Space List output JSON


```
{
  "data":
  [
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_001",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes",
    },
    {
      "uuu_rsv_overbook":"No overbooking",
```

```
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_002",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes",
    }
  ],
  "message":
  [ "success"
  ],
  "status":200
}
```


Status codes are:

1> 200 , for success

Create Space

POST /ws/rest/service/v1/space

Purpose:

To create Space record. This works only at project or shell .

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create Space Record input JSON


```
{
  "options" :
  {
      "project_number":"Shell/Project number",
      "space_type" : "Space BP name"
  },
  "data": [
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
```

```
      "space_sp_space_name":"Space_003",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    },
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_004",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    }
  ]
}
```

Input JSON field description:

- project_number (Required): specify the project/shell number to create space record.
- space_type (Required): specify the Space BP name, for which record will be created.
- space_sp_space_name (Required) : Unique "**space_sp_space_name**" must be provided in "data" for each space record.

Output:

JSON object containing '**status**', '**data**', '**message**'

Create Space Record output JSON

```
{
  "data": [
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_003",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
```

```
      "uuu_rsv_reservable":"Yes"
    },
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_004",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    }
  ],
  "message": [
    {
      "space_sp_space_name":"Space_003",
      "message":"success"
    },
    {
      "space_sp_space_name":"Space_004",
      "message":"success"
    }
  ],
  "status": 200
}
```


Status codes are:

1> 200 , for success.
2> 3000, for partial success.

Update Space

PUT /ws/rest/service/v1/space

Purpose:

To update Space record. This works only at project or shell .

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update Space Record input JSON

```
{
  "options" :
  {
      "project_number":"Shell/Project number",
      "space_type" : "Space BP name"
  },
  "data": [
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_003",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    },
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_004",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    }
  ]
}
```

Input JSON field description:

- project_number (Required): specify the project/shell number to update space record.
- space_type (Required): specify the Space BP name, where record will be Updated.
- space_sp_space_name (Required) : "**space_sp_space_name**" must be provided in "data" to identify space record to be updated.

Output:

JSON object containing '**status**', '**data**', '**message**'

Update Space Record output JSON


```
{
  "data": [
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_003",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    },
    {
      "uuu_rsv_overbook":"No overbooking",
      "uBuildingName":"Building_01",
      "space_sp_space_name":"Space_004",
      "uSiteName":"Site_01",
      "uuu_sp_level_picker":"floor_01",
      "uplRoomTB250":"sp_901",
      "uuu_rsv_reservable":"Yes"
    }
  ],
  "message": [
    {
      "space_sp_space_name":"Space_003",
      "message":"success"
    },
    {
      "space_sp_space_name":"Space_004",
      "message":"success"
    }
```

```
  ],
  "status": 200
}
```

Status codes are:

1> 200 , for success.

2> 3000, for partial success.

REST API Details in Cost

> **Note**: The integration user must have the required Cost Services Permissions.

In This Section

Get CBS Codes 299
Create CBS Codes 301
Update CBS Codes 304
Get Column Data 306
Update Column Data 307

Get CBS Codes

POST /ws/rest/service/v1/cost/cbs/list/{project_number}

Purpose:

Get list of CBS codes for a project number specified.

The input JSON includes options for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_num: Specify the Project number in which the records exists, Project number is mandatory

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body

```
{
"options":
{
"cost_type":"Capital",
"summary_detail":"true",
"status":"active"
"hierarchy":true
}
}
```

All the options specified are optional.

If hierarchy is not specified, default value is false. Complete hierarchy of the cbs code will be displayed if the value is true.

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

Note: Cost code attributes defined will be sent in JSON output response.

Get CBS Codes Sample Response


```
{
   "data":[
      {
         "bitemid" : 24,
         "code": "CostB",
         "description": "Desc of CostB",
         "item": " Cost Item",
         "uResProgressAmount": 10,
         "status": "Active",
         "cost_type": "Food",
         "parentid": "1",
         "orderid": "1",
         "budgetid": "6",
      },
      {
         "bitemid" : 25,
         "code": "CostB~~Cost1",
         "description": "Desc of CostB~Cost1",
         "item": " Cost Item",
         "uResProgressAmount": 10,
         "status": "Active",
         "cost_type": "Food",
         "parentid": "24",
```

```
            "orderid": "2",
            "budgetid": "6",
         },
         {
            "bitemid" : 26,
            "code": "Cost1",
            "description": "Desc of Cost1",
            "item": " Cost1 Item",
            "uResProgressAmount": 10,
            "status": "Active",
            "cost_type": "Capital",
            "parentid": "1",
            "orderid": "0",
            "budgetid": "6",
         },
        "codestructure": [
           {
               "treemode": true,
               "segsize": 1,
               "seperator": "-"
            }
        ],
    ],
    "message":[
       "success"
    ],
    "status":200
}
```


Create CBS Codes

POST /ws/rest/service/v1/cost/cbs/{project_number}

Purpose:

Create CBS Codes for the specified Project

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number. Project number is mandatory.

Both input & output in JSON format in the body

In the input request: "code" is Required field

Notes:

- If the CBS code is given in this format 'A~~B', then CBS code B will be created under A. If the cost code A does not exist, then the CBS code A will be created first, before creating the CBS code B under it.
- If the input code does not contain delimiter '~~', then that CBS code will be created under valid 'parentid'. If the 'parentid' is not available in the input request, then the code will be created under root.
- If the input code has delimiter '~~', and the 'parentid' is included in input request, then the 'parentid' will be ignored, and the code will be created as per hierarchy in the code.
- If multiple codes are sent in one request, and if the creation of one of the codes fails, then the entire request will be cancelled.

Create CBS code input JSON

```
{
    "data": [
        {
            "code": "CostB~~Cost1",
            "description": "Creating Cost Code using REST webservice",
            "item": " Cost Item",
            "uResProgressAmount": 10,
            "status": "Active",
            "cost_type": "Food"
        },
        {
            "code": "Cost1",
            "description": "Creating Cost Code using REST webservice",
            "item": "Cost1 Item",
            "uResProgressAmount": 10,
            "status": "Active",
            "cost_type": "Capital"
        }
```

```
    ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Create CBS Code output JSON


```
{
   "data":[
      {
         "bitemid" : 24,
         "code": "CostB",
         "description": "Creating Cost Code using REST webservice",
         "item": " Cost Item",
         "uResProgressAmount": 10,
         "status": "Active",
         "cost_type": "Food",
         "parentid": "23",
         "orderid": "1",
         "budgetid": "6",
      },
      {
         "bitemid" : 25,
         "code": "CostB~~Cost1",
         "description": "Creating Cost Code using REST webservice",
         "item": " Cost Item",
         "uResProgressAmount": 10,
         "status": "Active",
         "cost_type": "Food",
         "parentid": "24",
         "orderid": "2",
         "budgetid": "6",
      },
      {
         "bitemid" : 26,
```

```
            "code": "Cost1",
            "description": "Creating Cost Code using REST webservice",
            "item": " Cost1 Item",
            "uResProgressAmount": 10,
            "status": "Active",
            "cost_type": "Capital",
            "parentid": "1",
            "orderid": "0",
            "budgetid": "6",
         }
      ],
      "message":[
         "success"
      ],
      "status":200
}
```


Update CBS Codes

PUT /ws/rest/service/v1/cost/cbs/{project_number}

Purpose:

Update CBS Codes for the specified Project.

Input:

All parameters should be URL encoded.

Path Parameter

- project_number: Specify the Project number. Project number is mandatory.
- input query parameter -

  options=

  ```
  {
  "key": "bitemid"
  }
  ```
- Based on the key value provided in 'options' parameter, required field will be chosen. If key value is not given then 'code' will be considered as required field.
- "key" <<code>,<bitemid>>

  Notes:
  - The complete path from parent to child CBS code (separated by delimiter '~~') should be given in code field of input request if code is chosen as value for key.

- The "orderid" and "parentid" cannot be updated. All other cost attributes can be updated.
- The value in the CBS code field can be updated when the value of key in options parameter is"'bitemid."
- Partial Update is not allowed.

Update CBS code input JSON

```
{
    "data": [
        {
         "bitemid": "26",
         "code": "CostX"
         "description": "Updating Cost Code using REST webservice",
         "status": "InActive"
        }
    ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Update CBS Code output JSON

```
{
   "data":[
      {
         "bitemid" : 26,
         "code": "CostX",
         "description": "Updating Cost Code using REST webservice",
         "item": " Cost1 Item",
         "uResProgressAmount": 10,
         "status": "inactive",
         "cost_type": "Capital",
         "parentid": "1",
         "orderid": "0",
         "budgetid": "6",
      }
   ],
   "message":[
```

```
      "success"
   ],
   "status":200
}
```

Get Column Data

GET /ws/rest/service/v1/cost/columndata/{project_number}

Purpose:

Get column data of all rows / lineitems of costsheet for a given column name

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number to get column data of project cost sheet

Query Parameter

columnname(Required): Specify the column name of cost sheet.

Both input & output in JSON format in the body.

Output:

JSON object containing 'status', 'data', 'message'

Get Column Data ouptut JSON

```
{
  "data": [
    {
      "wbs_code": "Cost Code 1",
      "short_description": "Direct Entry Line Item",
      "amount": "1000.0",
      "spends_category": "",
      "quantity": "10.0",
      "work_package": "",
      "unit_of_measure": "each",
      "unit_cost": "100.0",
      "long_description": ""
    },
    {
```

```
      "wbs_code": "Cost Code 2",
      "short_description": "Direct Entry Line Item",
      "amount": "100.0",
      "spends_category": "",
      "quantity": "10.0",
      "work_package": "",
      "unit_of_measure": "",
      "unit_cost": "10.0",
      "long_description": ""
    }
  ],
  "message": [
    "success"
  ],
  "status": 200
}
```


Status codes are:

1> 200, for success

Update Column Data

PUT /ws/rest/service/v1/cost/columndata/{project_number}

Purpose:

Update column data of all rows / lineitems of costsheet for a given column name

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number to update column data of project cost sheet

Query Parameter

columnname(Required): Specify the column name of cost sheet.

Both input & output in JSON format in the body

Update Column Data input JSON


```
{
```

```
"data": [
{
  "wbs_code": "Code 1",
  "short_description": "Code 1 Description",
  "amount": "20.0",
  "spends_category": "",
  "quantity": "2.0",
  "work_package": "",
  "unit_of_measure": "each",
  "unit_cost": "10.0",
  "long_description": ""
},
{
  "wbs_code": "Code 2",
  "short_description": "Code 2 Description",
  "amount": "90.0",
  "spends_category": "",
  "quantity": "3.0",
  "work_package": "",
  "unit_of_measure": "fl. oz",
  "unit_cost": "30.0",
  "long_description": ""
},
{
  "wbs_code": "Code 3",
  "short_description": "Code 3 Description",
  "amount": "160.0",
  "spends_category": "",
  "quantity": "4.0",
  "work_package": "",
  "unit_of_measure": "",
  "unit_cost": "40.0",
  "long_description": ""
},
```

```
    {
      "wbs_code": "Code 3",
      "short_description": "Code 3 Description",
      "amount": "250.0",
      "spends_category": "",
      "quantity": "5.0",
      "work_package": "",
      "unit_of_measure": "yd.",
      "unit_cost": "50.0",
      "long_description": ""
    }
  ],
  "message": [
    "success"
  ],
  "status": 200
}
```


> **Note**: Data of all rows must be provided in the request otherwise their respective line items will be deleted and amounts of the rows which are not part of request will be reset to 0

Output:

JSON object containing 'status', 'data', 'message'

Update Column Data output JSON


```
{
    "data": [
    {
      "wbs_code": "Code 1",
      "short_description": "Code 1 Description",
      "amount": "20.0",
      "spends_category": "",
      "quantity": "2.0",
      "work_package": "",
      "unit_of_measure": "each",
      "unit_cost": "10.0",
```

```
    "long_description": ""
  },
  {
    "wbs_code": "Code 2",
    "short_description": "Code 2 Description",
    "amount": "90.0",
    "spends_category": "",
    "quantity": "3.0",
    "work_package": "",
    "unit_of_measure": "fl. oz",
    "unit_cost": "30.0",
    "long_description": ""
  },
  {
    "wbs_code": "Code 3",
    "short_description": "Code 3 Description",
    "amount": "160.0",
    "spends_category": "",
    "quantity": "4.0",
    "work_package": "",
    "unit_of_measure": "",
    "unit_cost": "40.0",
    "long_description": ""
  },
  {
    "wbs_code": "Code 3",
    "short_description": "Code 3 Description",
    "amount": "250.0",
    "spends_category": "",
    "quantity": "5.0",
    "work_package": "",
    "unit_of_measure": "yd.",
    "unit_cost": "50.0",
    "long_description": ""
```

```
    }
  ]
  "message": [
    "success"
  ],
  "status": 200
}
```

Status codes are:

1> 200, for success

REST API Details in Cashflow

> **Note**: The integration user must have the required Cashflow Services Permissions.

1) Properties (Project/Shell, Summary CBS, CBS, Commitment)
   a. Get Properties
   b. Create Properties
   c. Update Properties
2) Detail Curve (Baseline, Spends, Forecast, Custom, Derived, Portfolio Budget)
   a. Get Detail Curve
   b. Update Detail Curve
   c. Delete Curve
3) Refresh Curve
4) Get Templates
5) Schedule Refresh Curve
6) Add/update CF permission
7) GET Property by prop_name (one record)

In This Section

Definitions or Values Used 314
Response Error Codes (REST API Details in Cashflow) 318
Get Properties List 325
Get Templates List 340
Create Cashflow 347
Create Summary Curve 358
Create Rollup Cashflows for Program or Company 360
Update Rollup Cashflows for Program or Company 363
Delete Cashflow 366
Delete Cashflow - Summary Curves 367
Get Cashflow Data 368
Get Summary Cashflow Data 372
Get Rollup Status 376
Get Rollup Status - For Template Cashflows 377
Update Rollup Status 378
Update Rollup Status - For Template Cashflows 380
Refresh Cashflow 381
Get Cashflow Refresh Job Status 383
Get Cashflow Properties 384
Get Summary Cashflow Properties 409
Get Cashflow Permissions 410
Update or Modify Cashflow Permission 412
Create (Add User or Group) Cashflow Permission 418
Delete (Remove User or Group) Cashflow Permission 422
Update Cashflow 424
Update Cashflow Data 429
Create Distribution Profiles 431
Get Distribution Profiles 434
Update Distribution Profiles 436
Delete Distribution Profiles 439
Refresh Cashflow Curves 440
Get Cashflow Permissions 442

Definitions or Values Used

DE	Value	Label	Display UI Text
rollup_status	0	Active	Active
rollup_status	1	Inactive	Inactive
detail_level	0	Project / Shell	Project / Shell

detail_level	1	CBS	CBS
detail_level	2	Summary CBS	Summary CBS
detail_level	3	Commitment	Commitment
period_type	No	Standard Planning Period	Standard Planning Period
period_type	Yes	Financial Periods	Financial Periods
inc_spends_opt	0	next_month	The next month if after the cutoff date
inc_spends_opt	2	same_month	The same month as the effective date
distribution >type	0	manual	Manual
distribution >type	2	manual_autodistribution	manual auto distribute by default profile
distribution >type	3	schedule_sheet	Use data from Schedule Sheet
distribution >type	4	p6_sheet	Use data from P6 Source
Cost >Type	0	manual	Manually enter amounts for each period
Cost >Type	1	auto_distribution	Auto distribute total amount across all periods
Cost >Type	2	cost_sheet	Distribute amount from cost sheet column
Cost >Type	3	p6_sheet	Use Actuals from P6

			Summary Sheet
Schedule >Type	0	manual	Manual
Schedule >Type	1	schedule_sheet	Use dates from Schedule Sheet
Schedule >Type	2	p6_sheet	Use dates from P6 Source
Schedule >Type	3		
Filter >Type	0	all	All CBS codes from cost sheet
Filter >Type	1	summary_cbs	Select summary CBS codes
Filter >Type	2	cbs	Select CBS codes
Forecast > Option	TRUE	begin_end_of_curve	Begin calculations at end of curve
Forecast > Option	TRUE	current_actual_cut_off_date	Replace current period forecast with Actuals on cut Off date
Distribute unassigned amounts from Spends			
Distribute unassigned amounts from Spends	0	weighted_avg_all_periods	Using weighted average over all remaining periods

Derived > Float Rate Option			
Derived > Float Rate Option	0	beginning_period	At the beginning of the period
Derived > Float Rate Option	1	end_period	At the end of the period
Derived > Float Rate Option	2	weighted_period	A weighted average for the period
Detail level > Commitment Granularity	0	single_distribution	Single distribution of sum of base commit and all changes
Detail level > Commitment Granularity	1	individual_record	By individual record(base commit and all changes separately)
Detail level > Commitment Granularity	2	by_lineitem	By line items within each individual record
Distribution method	0	manual	
Distribution method	1	Autodistribution_manual	Autodistribution - manual dates and profile
Distribution method	1	Autodistribution_bp {bp:"",from:"",to:"",profile:""}	Autodistribution - BP dates and default profile selection
Commit change Processes		BP : Status	

Response Error Codes (REST API Details in Cashflow)

Code	Message	Details
200	success	If the request is successful
5000	Server Error, contact the system administrator.	If there is any exception when running this service
505	token+" is not present, it is required in the formula:"+value+" for field "+name	If the data provided is not correct like pull down with a label which does not exist.
602	Project/Shell Number is not correct	project number is incorrect or does not exists
603	Business Process Name is not correct	"bpname" is not specified in the input JSON and or is invalid value
651	CBS Code is not present.	"cbs_code" is not present in the input JSON
657	Invalid record_no.	"record_no" value is not specified in the input JSON or is invalid value.
663	Status field value is not correct.	If status value is not valid i.e it is neither active nor inactive
701	Invalid Column Name	If column name given in the filter condition does not belong to funding sheet
11001	Empty or Invalid JSON data	If required input JSON is empty
13001	Cash Flow name is not present in input request	

13002	Cash Flow Template name provided is not valid	
13003	Cash Flow data provided is not valid for field :	If invalid data is provided in input JSON for fields
13004	Required Cash Flow data is not provided for field :	If data is not provided in input JSON for mandatory fields
13005	Cash Flow do not exist	Cashflow record with given property name does not exist
13006	System-defined Summary Cash Flow Curve cannot be deleted	
13007	Cash Flow not Found	
13008	Summary Cash Flow Curve name already exists	If the name already exists when creating cashflow summary
13009	More than one Cash Flow Curve is required to create a Summary Cash Flow Curve	
13010	Error in creating Summary Cash Flow Curve	
13111	Created Summary Cash Flow Curve without these invalid Cash Flow Curve(s):	
13111	Cannot create Summary Cash Flow Curve since all Cash Flow Curve(s) are invalid	

2		
1313	Summary Cash Flow Curve name is required	
1314	Cash Flow name provided already exists.	
1315	Cost sheet is required for detail_level CBS/Summary CBS/Commitment.	
1316	Cost sheet is required for Summary CBS/CBS filters.	
1317	Financial Period is not assigned to the shell.	
1318	Filter Type Summary CBS can only be assigned to Tree mode of Cost sheet.	
1319	CashFlow Template name is required.	
1320	Request is valid for Shell or Project	If the request is not valid for program or company.
1321	Template Number is not correct.	
1322	Invalid Job ID.	
13	Job processing failed with errors.	

23		
13244	Detail curve is not present in the CashFlow	when performing get data for a cashflow which doesn't have a detail curve.
13255	This operation is not supported on Inactive or View-only Shell/Project/Program	
13266	Detail curves cannot be created for Program/Template	
13277	Summary Cash Flow status Should be 'Active' or 'Inactive'.	
13288	Request is not valid for shell template.	
13299	<different validation error for different fields in input JSON>	If fields provided in input JSON fails validation.
13300	Cash Flow Curve do not exist	If Update/Delete curve request input JSON contains curve details which does not exists on given CashFlow
13311	Error in deleting Cash Flow Curve	
13322	No Curves are defined for this Cash Flow	If Update/Delete curve request input JSON contains curve details for CashFlow which does not have any curve .
13333	Rollup Cash Flow name is required	Name is a required field in Rollup curve creation
1	Request is not valid for Program	

334		
1335	Rollup curves can be created/updated only for Company/Program	Rollup curves can be created only for company/program
1336	Detail curves cannot be updated for Program/Template	
1337	Distribution Profile do not exist	
1338	Distribution Profile is used in Cashflows Curves	
1339	filter parameter is required	For GET cashflow data & summary cashflow data , filter parameter is required.
1340	filter parameter is invalid	If filter condition is invalid for GET Cashflow data
1341	Error in updating the Cash Flow Data	
1342	Value for Number is not provided in the input data	
1343	Value for No is not provided in the input data	Value for Line Number is not provided in the input data
1344	Number or No has invalid value in the input data	

13445	Date values provided in the input is invalid	
13446	Profile name is invalid in the input request	
13447	Total value is invalid in the input request	
13448	Update data is not allowed	
13449	Cash Flow Curve name is not present in input request	
13450	Invalid CBS Code	
13451	From date is not set	
13452	To date is not set	
13453	Invalid distribution amount	
13454	Provided Currency is not a valid transaction currency	
13455	Duplicate data exists in details	

5		
13556	Cash Flow operations are not possible on Generic type of Project	
13557	Rollup Cash Flow Curve does not exist	
13668	Create/update request contains conflicting values for noenddate and enddate	
13559	Cash Flow id is required	
13600	Cash Flow Curve details are not present in the input request	If input JSON for update cashflow data contains curves but curve details are not provided
13611	Project number is required	This error is raised , If Cashflow data update REST call is for Company level
13622	Detail level Summary CBS cannot have filters of type CBS	
13633	Detail level CBS cannot have filters of type Summary CBS	
13644	GET Detail curves operation is not supported for Program/Template/Company	
13655	From Date should be greater than Financial Period Start Date and Less than Financial Period End Date	
13	To Date should be less than Financial Period End Date and	

67	Greater than Financial Period Start Date	
1368	Cannot update distribution as dates are not defined for this Cash Flow curve	
1369	No data are updated as both setup and distribution options are not set	
1370	Cash Flow data provided cannot be updated for the field :	
30000	Partial failure.	Partial failure of REST service
30002	Invalid input	If input JSON didn't contain the "data"/required parameter
30003	Business Process is not Active.	When fund consumption is performed on a inactive BP.

Get Properties List

POST /ws/rest/service/v1/cashflow/prop/list/{project_number}

Purpose:

Get all list of CashFlow properties with all curves types defined in CashFlow.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: (Required) Specify the Project number in which the curve exists.

POST body is a JSON

Note: POST call has input & output both as JSON in the body.

```
{
"options":{
"property_name": "CBS CF"
}
}
```

Supported options properties

"property_name": "CBS CF"

"curve_name": << Name of The Curve>>,

"rollup_status"<<Active><Inactive>>

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_levelvalue provided is other than above mentioned values,invalid options message will be thrown.

"include_curves":<< Yes/ No>> <!-- default is Yes if not provided>

If include_curves value provided is other than above mentioned values, invalid options message will be thrown.

If no options provided service will return all templates along with cashflow curves from the project.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Project or Shell Sample

```
{
    "data": [
        {
            "name": "Project_CF",
            "description": "test webservice get Cashflow Properties",
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
```

```
"decimal_places": 8,
"snapshot": {
    "snapshot_day_of": 1,
    "cutoff_spends": {
        "cutoff_week_num": 2,
        "cutoff_week_day": "4",
        "inc_spends_opt": "next_month"
    }
},
"curves": [
    {
        "name": "Baseline",
        "type": "Baseline",
        "distribution": {
            "type": "auto_profile",
            "distribution_profile": "Front Loaded"
        },
        "cost": {
            "type": "auto_distribution"
        },
        "schedule": {
            "type": "schedule_sheet",
            "sheet_name": "sheet3",
            "sch_start_source": "Estimated Finish Date",
            "sch_end_source": "Estimated Start Date"
        }
    },
    {
        "name": "Forecast",
        "type": "Forecast",
        "distribution": {
            "type": "manual"
        },
        "cost": {
```

```
                "type": "manual"
            },
            "schedule": {
                "type": "schedule_sheet",
                "sheet_name": "Sheet1",
                "sch_start_source": "AP_Date6",
                "sch_end_source": "AP_Date only 1"
            },
            "forecast_options": {
                "begin_end_of_curve": true,
                "fc_curve_name": "Actuals",
                "current_actual_cut_off_date": true,
                "edit_until_replace_by_actual": true
            }
        },
        {
            "name": "Actuals",
            "type": "Spends",
            "cost": {
                "cost_sheet_column": "Purchase Orders_Pending",
                "type": "cost_sheet"
            }
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "United States Dollar (USD)",
                "distribut_amount_from": "original"
            }
        },
        {
            "name": "Approved Budget",
            "type": "Portfolio Budget",
```

```
                "cost": {
                    "currency_code": "RADO",
                    "distribut_amount_from": "approved"
                }
            },
            {
                "name": "Shared Budget",
                "type": "Portfolio Budget",
                "cost": {
                    "currency_code": "Afghani (AFN)",
                    "distribut_amount_from": "shared"
                }
            },
            {
                "name": "custom_D",
                "type": "Derived",
                "cost": {
                    "base_currency": "United States Dollar (USD)",
                    "source_curve_name": "Baseline",
                    "derived_curve_currency": "Afghani (AFN)",
                    "exchange_rate": "Float",
                    "float_rate_as": "end_period"
                }
            }
        ],
        "filters": {
            "filter_option": "all"
        }
    }
],
"message": [
    "success"
],
"status": 200
```

```
}
```

CBS Sample

```
{
    "data": [
        {
            "name": "CBS_CF",
            "description": "CBS  Cashflow",
            "rollup_status": "Active",
            "detail_level": "CBS",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Year",
            "period_format": "YYYY",
            "decimal_places": 3,
            "snapshot": {
                "snapshot_week_num": 1,
                "snapshot_week_day": "1"
            },
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "cost_sheet_column": "Purchase Orders_Pending",
                        "type": "cost_sheet"
                    }
                },
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
                        "type": "auto_profile",
                        "cbs": [
```

```
                            {
                                "cbs_code": "s1-a 1-b 1-a123~~s3-a 2-b
1-a123",
                                "bitemid": 13,
                                "item": "A123",
                                "default_profile": "Front Loaded"
                            },
                            {
                                "cbs_code": "s1-a 1-b 1-a123~~s2-a 2-b
1-a123",
                                "bitemid": 12,
                                "item": "A123",
                                "default_profile": null
                            }
                        ]
                    },
                    "cost": {
                        "type": "auto_distribution"
                    },
                    "schedule": {
                        "type": "schedule_sheet",
                        "sheet_name": "Sheet1",
                        "sch_start_source": "Finish date",
                        "sch_end_source": "Late Start date"
                    }
                },
                {
                    "name": "Forecast",
                    "type": "Forecast",
                    "distribution": {
                        "type": "schedule_sheet",
                        "sch_sheet_name": "sheet2"
                    },
                    "cost": {
```

```
                        "type": "manual"
                    },
                    "schedule": {
                        "type": "manual"
                    },
                    "forecast_options": {
                        "begin_end_of_curve": true,
                        "fc_curve_name": "Actuals",
                        "current_actual_cut_off_date": true,
                        "fc_dist": "weighted_avg_over_next",
                        "periods": 1
                    }
                },
                {
                    "name": "Protfolio Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Costa Rica Colon (CRC)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "custom_D",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Float",
                        "float_rate_as": "weighted_period"
                    }
                },
                {
```

```
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Bhutan Ngultrum (BTN)",
                "distribut_amount_from": "approved"
            }
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "RADO",
                "distribut_amount_from": "original"
            }
        },
        {
            "name": "Shared Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "RADO",
                "distribut_amount_from": "shared"
            }
        },
        {
            "name": "Derive",
            "type": "Derived",
            "cost": {
                "base_currency": "United States Dollar (USD)",
                "source_curve_name": "Forecast",
                "derived_curve_currency": "Dirham (AED)",
                "exchange_rate": "Float",
                "float_rate_as": "beginning_period"
            }
        },
```

```
                {
                    "name": "Forecast 2",
                    "type": "Forecast",
                    "distribution": {
                        "type": "manual"
                    },
                    "cost": {
                        "type": "manual"
                    },
                    "schedule": {
                        "type": "p6_sheet",
                        "p6_datasource": "Original Baseline",
                        "sch_dates_type": "At Completion"
                    },
                    "forecast_options": {}
                }
            ],
            "filters": {
                "filter_option": "all"
            }
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Summary CBS Sample


```
{
    "data": [
        {
            "name": "SummaryCBS_CF",
            "description": "SummaryCBS_CF for Cashflow Properties",
```

```
"rollup_status": "Active",
"detail_level": "Summary CBS",
"period_type": "Standard Planning Period",
"period_name": "Standard Planning Period",
"period_by": "Month",
"period_format": "M YYYY",
"decimal_places": 3,
"snapshot": {
    "snapshot_week_num": 1,
    "snapshot_week_day": "2",
    "cutoff_spends": {
        "cutoff_week_num": 2,
        "cutoff_week_day": "6",
        "inc_spends_opt": "next_month"
    }
},
"curves": [
    {
        "name": "Actuals",
        "type": "Spends",
        "cost": {
            "cost_sheet_column": "Purchase Orders_Pending",
            "type": "cost_sheet"
        }
    },
    {
        "name": "Baseline",
        "type": "Baseline",
        "distribution": {
            "type": "auto_profile"
        },
        "cost": {
            "type": "cost_sheet",
            "cost_sheet_column": "Purchase Orders (Pending)"
```

```
            },
            "schedule": {
                "type": "schedule_sheet",
                "sheet_name": "Sheet1",
                "sch_start_source": "Start date",
                "sch_end_source": "Finish date"
            }
        },
        {
            "name": "Forecast",
            "type": "Forecast",
            "distribution": {
                "type": "manual"
            },
            "cost": {
                "type": "manual"
            },
            "schedule": {
                "type": "manual"
            },
            "forecast_options": {}
        },
        {
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "test",
                "distribut_amount_from": "approved"
            }
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
```

```
                        "currency_code": "Yen (JPY)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Algeria Dinar (DZD)",
                        "distribut_amount_from": "shared"
                    }
                },
                {
                    "name": "Derive",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Baseline",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Float",
                        "float_rate_as": "beginning_period"
                    }
                }
            ],
            "filters": {
                "filter_option": "all"
            }
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
```

```
}
```

Commitment Type Sample

```
{
    "data": [
        {
            "name": "Commitment_CF",
            "description": "commit",
            "rollup_status": "Active",
            "detail_level": "Commitment",
            "bp_name": "Purchase Orders",
            "reference_elements": "record_no:Record No.",
            "base_commit_record": "PO-0001",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cashflow_granularity": false,
                    "cost": {
                        "cost_sheet_column": "{ Invoices:Approved }{
Invoices:Canceled }{ Invoices:Pending }{ Invoices:Rejected }"
                    }
                },
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "cashflow_granularity": "individual_record",
                    "distribution": {
                        "type": "auto_profile",
```

```
                        "commit_profile": "business_process",
                        "bp_dates_profile": "{ Purchase Orders:Creation
Date:Creation Date:Back Loaded }{ PO Amendments:Creation Date:Creation
Date:Linear }"
                    },
                    "commit_change_processes": "{PO
Amendments:Approved}{PO Amendments:Canceled}{PO Amendments:Rejected}"
                },
                {
                    "name": "Forecast",
                    "type": "Forecast",
                    "cashflow_granularity": "by_lineitem",
                    "distribution": {
                        "type": "manual"
                    },
                    "commit_change_processes": "",
                    "forecast_options": {}
                },
                {
                    "name": "Approved Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Costa Rica Colon (CRC)",
                        "distribut_amount_from": "approved"
                    }
                },
                {
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Dirham (AED)",
                        "distribut_amount_from": "shared"
                    }
                },
```

```
                {
                    "name": "Original Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Afghani (AFN)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Derive",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Baseline",
                        "derived_curve_currency": "Costa Rica Colon
(CRC)",
                        "exchange_rate": "Float",
                        "float_rate_as": "weighted_period"
                    }
                }
            ]
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Get Templates List

POST /ws/rest/service/v1/cashflow/template/list/{project_number}

Purpose:

Get all list of CashFlow Templates with all curves types defined.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the curve exists.

POST body is a JSON

Note: POST call has input & output both as JSON in the body.

```
{
"options":{
"property_name": "CBS CF"
}
}
{
   "options":{
      "names":[
         "CBS CF"
      ]
   }
}
```

Supported options properties

"property_name": "CBS CF"

“names": "CBS CF" for one cashflow

"names": ["cashflow_1", "cashflow_2"] for one or more cashflows

"curve_name": << Name of The Curve>>,

"rollup_status"<<Active><Inactive>>

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_levelvalue provided is other than above mentioned values,invalid options message will be thrown.

"include_curves":<< Yes/ No>> <!-- default is Yes if not provided→>

If include_curves value provided is other than above mentioned values, invalid options message will be thrown.

If no options provided service will return all templates along with cashflow curves from the project.

,

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Project or Shell Sample


```
{
    "data": [
        {
            "name": "Project Cash Flow - Capital Planning",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "curves": [
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
                        "type": "auto_profile",
                        "distribution_profile": "S Curve"
                    },
                    "cost": {
                        "type": "cost_sheet",
                        "cost_sheet_column": "Revised Budget"
                    },
                    "schedule": {
                        "type": "manual"
                    }
                },
                {
```

```
            "name": "Actuals",
            "type": "Spends",
            "cost": {
                "cost_sheet_column": "Spends",
                "type": "cost_sheet"
            }
        },
        {
            "name": "Forecast",
            "type": "Forecast",
            "distribution": {
                "type": "auto_profile",
                "distribution_profile": "S Curve"
            },
            "cost": {
                "type": "cost_sheet",
                "cost_sheet_column": "Forecast"
            },
            "schedule": {
                "type": "manual"
            },
            "forecast_options": {}
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "United States Dollar (USD)",
                "distribut_amount_from": "original"
            }
        },
        {
            "name": "Approved Budget",
            "type": "Portfolio Budget",
```

```
                    "cost": {
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "approved"
                    }
                },
                {
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "shared"
                    }
                }
            ],
            "filters": {
                "filter_option": "all"
            }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
{
    "data": [
        {
            "id": 36,
            "name": "Project Cash Flow - Capital Planning",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
```

```
"period_format": "M YYYY",
"decimal_places": 2,
"schedule": {
    "enable_refresh": false
},
"curves": [
    {
        "id": 148,
        "name": "Baseline",
        "type": "Baseline",
        "distribution": {
            "type": "auto_profile",
            "distribution_profile": "S Curve"
        },
        "cost": {
            "type": "cost_sheet",
            "cost_sheet_column": "Revised Budget"
        },
        "schedule": {
            "type": "manual"
        }
    },
    {
        "id": 149,
        "name": "Actuals",
        "type": "Spends",
        "cost": {
            "cost_sheet_column": "Spends",
            "type": "cost_sheet"
        }
    },
    {
        "id": 150,
        "name": "Forecast",
```

```
            "type": "Forecast",
            "distribution": {
                "type": "auto_profile",
                "distribution_profile": "S Curve"
            },
            "cost": {
                "type": "cost_sheet",
                "cost_sheet_column": "Forecast"
            },
            "schedule": {
                "type": "manual"
            },
            "forecast_options": {}
        },
        {
            "id": 151,
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "United States Dollar (USD)",
                "distribut_amount_from": "original"
            }
        },
        {
            "id": 152,
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "United States Dollar (USD)",
                "distribut_amount_from": "approved"
            }
        },
        {
            "id": 153,
```

```
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "shared"
                    }
                }
            ],
            "filters": {
                "filter_option": "all"
            }
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Create Cashflow

POST /ws/rest/service/v1/cashflow/{project_number}

Purpose:

Create Cashflow from a template or by specifying the property values.

If the template name is provided, cashflow properties are created from a template in combination with input parameters specified in the request.

If template_projectid is not provided, the company template will be fetched.

Integration user used will have full access to the cashflow created. To change the permissions, **Update or Modify Cashflow Permission** service need to be run.

In addition to integration user, project administrator/company administrator will have full access to the cashflow created from service.

Period name is not accepted from the input JSON. If provided, will be ignored.

Irrelevant attributes specified in the request for create/update are ignored.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Required

When creating cashflow from template specify the below values in the data JSON

template_name:Required

template_project: optional. Project number of the template.

Post data JSON will be details of cashflow

For the input attributes provided, the data will be used from the input request -

Example

If name, template, and schedule details are specified in the input request, then the cashflow will be created with template details but the schedule will be as per the input request.

Defaults will be used if data is not provided in the input request.

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Create Cashflow From Template Only

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

}
```

Create Cashflow Manual - Project or Shell

```
{
    "data":
        {
            "name": "project type cashflow",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
```

```
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
           "snapshot": {
                "sn_week_num": "First",
                "sn_week_day": "Sunday",
                "cutoff_spends": {
                    "cutoff_week_num": "First",
                    "cutoff_week_day": "Sunday",
                    "inc_spends_opt": "same_month"
                }
            },
            "schedule": {
                "enable_refresh": false
            },

            "summary": [
                {
                    "name": "summary",
                    "summary_type": "User_Defined"
                }
            ],
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "type":"cost_sheet",
                        "cost_sheet_column": "152 Base Commit WF
(Approved)"
                    }
                },
                {
                    "name": "Cash flow3",
```

```
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Australia Dollar
(AUD)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Dynamic",
                        "peg_dynamic_ds": "Single Record BPs / 922_Simple
PNW/FSingle LL",
                        "peg_dynamic_de": "Currency amount 1",
                        "float_rate_as": "weighted_period"
                    }
                },
                {
                    "name": "Today",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Australia Dollar (AUD)",
                        "distribut_amount_from": "approved"
                    }
                },
                {
                    "name": "Forecast",
                    "type": "Forecast",
                    "distribution": {
                        "type": "auto_profile",
                        "distribution_profile": "Front Load"
                    },
                    "cost": {
                        "type": "cost_sheet",
                        "cost_sheet_column": "152 Base Commit WF (Open)"
                    },
                    "schedule": {
```

```
                    "type": "manual"
                },
                "forecast_options": {
                    "begin_end_of_curve": true,
                    "fc_curve_name": "Actuals",
                    "current_actual_cut_off_date": true,
                    "fc_dist": "weighted_avg_over_next",
                    "periods": 3
                }
            }
        ]
    }

}
```

Create Cashflow Manual - CBS

```
{
    "data":
        {
           "name": "cbsfromservice",
            "description": "cbs type",
            "rollup_status": "Active",
            "detail_level": "CBS",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "snapshot": {
                "sn_dayof": 1,
                "cutoff_spends": {
                    "cutoff_week_num": "First",
                    "cutoff_week_day": "Sunday",
                    "inc_spends_opt": "same_month"
```

```
                }
            },
        "filters": {
                "filter_option": "cbs",
                "cbs": [
                    {
                        "cbs_code": "16000~~16700",
                        "bitemid": 235,
                        "item": "Communications"
                    },
                    {
                        "cbs_code": "99999",
                        "bitemid": 237,
                        "item": "Contingencies"
                    }
                ]
            },
            "summary": [
                {
                    "name": "Cash Flow Summary Curve",
                    "summary_type": "System_Defined"
                }
            ],
            "curves": [
                {
                    "name": "P6Actuals",
                    "type": "Spends",
                    "cost": {
                        "type":"P6 Summary Sheet",
                        "p6_datasource":"Original Baseline" //"Current
Schedule"
                        "cost_sheet_column": "A_Commit_01 (Approved)"
                    }
                },
```

```
                {
                    "name": "custom_Float",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Float",
                        "float_rate_as": "weighted_period"
                    }
                },
                {
                    "name": "Cash flow3",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Australia Dollar
(AUD)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Dynamic",
                        "peg_dynamic_ds": "Single Record BPs / 922_Simple
PNW/FSingle LL",
                        "peg_dynamic_de": "Currency amount 1",
                        "float_rate_as": "weighted_period"
                    }
                },
                {
                    "name": "custom_Peg_Constant",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
```

```
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Constant",
                        "peg_constant_value":"1"
                    }
                },
                {
                    "name": "custom_Peg_Project",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Project"
                    }
                },
                {
                    "name": "Today",
                    "type": "Portfolio Budget",
                    "cost": {
                  "currency_code": "Indian Rupee (INR)",
                  "distribut_amount_from": "approved"
               }
            }
         ]
     }

}
```

Create Cashflow Manual - Summary CBS

```
{
```

```
    "data":
        {
            "name": "Manual Curve",
            "detail_level": "CBS",
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "type":"cost_sheet",
                        "cost_sheet_column": "152 Base Commit WF
(Approved)"
                    }
                },
                {
                    "name": "Today",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Australia Dollar (AUD)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Cash flow1",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Today",
                        "derived_curve_currency": "Afghani (AFN)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Dynamic",
                        "peg_dynamic_ds": "Single Record BPs / 922_Simple
PNW/FSingle LL",
                        "peg_dynamic_de": "Integer Amount 1",
```

```
                        "float_rate_as": "weighted_period"
                    }
                }
            ]
        }

}
```

Create Cashflow Manual - Commitment

```
{
    "data":
        {
            "name": "commitment",
            "description": "commitment",
            "rollup_status": "Active",
            "detail_level": "Commitment",
            "bp_name": "Purchase Orders",
            "reference_elements": "record_no:upoPONumberTXT16",
            "base_commit_record": "PO-0001",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
             "schedule": {
                "enable_refresh": true,
                "frequency": "Weekly",
                "weekday": "Sunday",
                "startdate": "06/10/2019 12:00 AM",
                "noenddate": false,
                "enddate": "06/24/2019 12:00 AM"
            },
            "summary": [
                {
```

```
                    "name": "Cash Flow Summary Curve",
                    "summary_type": "System_Defined"
                },
                {
                    "name": "summary",
                    "summary_type": "User_Defined"
                }
            ]
        }
            "curves": [
                {
                    "name": "commitActuals",
                    "type": "Spends",
                    "cashflow_granularity": "by_lineitem", // "" or
"single_distribution"
                    "cost": {
                        "spends_bp": [{"bp_name" :"A_Commit_01",
"status":"Approved"}]
                    }
                },
                {
                    "name": "Today",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Ariary (MGA)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Cash flow1",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Spend",
```

```
                        "derived_curve_currency": "Chile Peso (CLP)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Dynamic",
                        "peg_dynamic_ds": "Single Record BPs /
91ProjectinformationBP",
                        "peg_dynamic_de": "% Phase Cmplete",
                        "float_rate_as": "end_period"
                    }
                }
            ]
        }

}
```

Create Summary Curve

POST /ws/rest/service/v1/cashflow/summary/{project_number}

Purpose:

Create Cashflow summary curves.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number.

Note: POST call has input & output both as JSON in the body

In the input request:

"name"(Required).

"status" (Optional): If not specified then the summary curve will be active.

"description" (Optional): If no value is specified then the summary curve will be created with no description.

"curves" (Optional): List of cash flow curve names, which has to be part of the summary curve. If no value is specified then the summary curve will be created without any cash flow curve.

Output:

JSON object containing 'status', 'data', 'message'

The data will contain the name and the id of the summary curve which was created successfully. The message will contain the status of the curve creation for all the input data.

Sample input request


```
{
   "data":[
      {
         "name":"summary curve 1",
         "status":"active",
         "description":"This summary curve is created from REST API",
         "curves":[
            "CF For buildings",
            "Curve 1"
         ]
      },
      {
         "name":"summary curve 2",
         "status":"active",
         "description":"This summary curve is created from REST API",
         "curves":[
            "Manual curve 0120",
            "cashflow for bridge 20"
         ]
      }
   ]
}
```


Sample output request


```
{
   "data":[
      {
         "name":"summary curve 1",
         "id":31
      }
   ],
   "message":[
      {
```

```
            "name":"summary curve 1",
            "status":"200",
            "message":"success"
        },
        {
            "name":"summary curve 2",
            "status":"1308",
            "message":"Summary name already exists"
        }
    ],
    "status":3000
}
```


Create Rollup Cashflows for Program or Company

POST /ws/rest/service/v1/cashflow/rollup/{program_number}

Purpose:

Create Rollup Cashflows.

Input:

All parameters should be URL encoded.

Path Parameter

program_number(Optional): Specify the program number to create rollup cashflows in program otherwise to create in company

Both input & output in JSON format in the body

In the input request:

"template" (Optional): Is valid only for programs, if template number is given for company then that value will be ignored

"name" (Required)

"status" (Optional): If not specified then by default 'Active' will be taken.

"data_source" (Optional): List of data sources, which has to be part of the rollup curve. If no value is specified then the rollup curve will be created without any data sources.

"period_type" (Required): If no value is given then by default "Standard Planning Period" will be taken

"period_by": Is Required if period_type is "Standard Planning Period" and by default 'Year' will be taken.

"period_format": Is Required if period_type is "Standard Planning Period" and by default 'YYYY' will be taken.

"period_name": Is Required if period_type is "Financial Periods"

"decimal_places" (Optional)

Output:

JSON object containing 'status', 'data', 'message'

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.

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Create Rollup Cashflows Input JSON

```
{
    "data": [
        {
            "template": "Rollup Template"
            "name": "Rollup Curve 1",
            "status": "Active",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Year",
            "period_format": "YYYY",
            "decimal_places": 5,
            "data_source": [
                {
                    "name": "Baseline",
                    "curve_type": "Baseline"
                },
                {
                    "name": "Vendor Approved Budget",
                    "curve_type": "Baseline"
                }
            ]
        },
        {
            "name": "Rollup Curve 3",
```

```
            "status": "Active",
            "period_type": "Financial Periods",
            "period_name": "FP",
            "decimal_places": 5,
            "data_source": []
        }
    ]
}
```

Create Rollup Cashflows Output JSON

```
{
   "data":[
      {
         "id": "208"
         "name": "Rollup Curve 1",
         "status": "Active",
         "period_type": "Standard Planning Period",
         "period_name": "Standard Planning Period",
         "period_by": "Year",
         "period_format": "YYYY",
         "decimal_places": 5,
         "data_source": [
             {
                "name": "Baseline",
                "curve_type": "Baseline"
             },
             {
                 "name": "Vendor Approved Budget",
                 "curve_type": "Baseline"
             }
          ]
      }
   ],
   "message":[
```

```
      {
         "name":"Rollup Curve 1",
         "status":"200",
         "message":"success"
      },
      {
         "name":"Rollup Curve 3",
         "status": 1303,
         "message": "CashFlow data provided is not valid for field :
period_name"

      }
   ],
   "status":3000
}
```


Update Rollup Cashflows for Program or Company

PUT /ws/rest/service/v1/cashflow/rollup/{program_number}

Purpose:

Update Rollup Cashflows.

Input:

All parameters should be URL encoded.

Path Parameter

program_number(Optional): Specify the program number to update rollup cashflows of program otherwise to update rollup cashflows of company

Both input & output in JSON format in the body

In the input request:

"id" is a required field

Output:

JSON object containing 'status', 'data', 'message'

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.

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Update Rollup Cashflows Input JSON

```
{
    "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,
            "data_source": [
                    {
                        "name": "Baseline"
                    },
                    {
                        "name": "Vendor Approved Budget",
                        "action": "delete"
                    }
            ]
        }
    ]
}
```

Update Rollup Cashflows Output JSON

```
{
```

```
"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"
          },
          {
              "name": "Vendor Approved Budget",
              "curve_type": "Baseline"
          }
       ]
   },
   {
      "id": 198,
      "name": "Rollup Curve 2",
      "description": "description",
      "status": "Active",
      "period_type": "Financial Periods",
      "period_name": "FP1",
      "period_by": "Custom",
      "decimal_places": 5,
      "data_source": [
          {
             "name": "Baseline",
             "curve_type": "Baseline"
```

```
            }
         ]
      }
   ],
   "message":[
      {
         "name":"Rollup Curve 1",
         "status":"200",
         "message":"success"
      },
      {
         "name":"Rollup Curve 2",
         "status": "200",
         "message": "success"

      }
   ],
   "status":200
}
```


Delete Cashflow

DELETE /ws/rest/service/v1/cashflow/{project_number}

Purpose:

Delete detail/rollup cashflows

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Optional): Specify the project/shell/program number to delete the cashflows of project otherwise to delete the company rollup curves

Both input & output in JSON format in the body

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Delete Cashflow Input JSON

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

Delete Cashflow Ouput JSON

```
{
    "data": [],
    "message": [
        {
            "status": 200,
            "message": "success",
            "name": "Cash Flow 1"
        },
        {
            "status": 1305,
            "message": "Cash Flow do not exist",
            "name": "Cash Flow 2"
        }
    ],
    "status": 3000
}
```

Delete Cashflow - Summary Curves

DELETE /ws/rest/service/v1/cashflow/summary/{project_number}

Purpose:

Delete summary cashflow

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number to delete the summary cashflows of project

Both input & output in JSON format in the body

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Delete Cashflow Input JSON

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

Delete Cashflow Ouput JSON

```
{
    "data": [],
    "message": [
        {
            "status": 200,
            "message": "success",
            "name": "Summary Cash Flow 1"
        },
        {
            "status": 1306,
            "message": "System-defined Summary Cash Flow Curve cannot be
deleted",
            "name": "Cash Flow Summary Curve"
        }
    ],
    "status": 3000
}
```

Get Cashflow Data

GET /ws/rest/service/v1/cashflow/data/{project_number}

Purpose:

Get Cashflow curve data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Optional): Specify the project/shell number. If no value is specified then company level cash flow data will be retrieved.

Url Parameter

filter (Required): Filter condition to be used when retrieving the data. It is a JSON object with the below information:

"name"(Required): Specify the cashflow name to get the data

"curves"(Optional): Specify the curve name for which the data is to be retrieved. Invalid curve name will be ignored. If no value is specified then all the curves data will be sent in the response.

Sample filter parameter

```
filter={  "name":"cashflow 1",   "curves":[  "Baseline", "Forecast"  ] }
```

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK if success

For Partial failure, the relevant message will be displayed.

Sample output request


```
{
   "data":[
      {
         "name":"cashflow 1",
         "alert_message":[
            {
               "projectname":"Shell-Weekly",
               "name":"CF-New",
               "curvename":"Actuals",
               "errormessage":"Curve timescale is yearly"
            },
            {
               "projectname":"SUNEEL-RETEST-YEARLY",
               "name":"CashFlow-Suneel-Retest-Yearly",
```

```
            "curvename":"Actuals",
            "errormessage":"Curve timescale is yearly"
         }
      ],
      "curves":[
         {
            "Name":"Baseline",
            "alert_message":"",
            "Currency":"Indian Rupee (INR)",
            "Type":"Baseline",
            "From Date":"01/01/17",
            "To Date":"12/30/17",
            "Total":"1742.56"
            "Unassigned":"300.90",
            "2 2016":"800.56"
            "details":[
               {
                  "Number":"A0001-B0001-C0001-D0001-E0001",
                  "Name":"Row-00001",
                  "Profile":"Front Load",
                  "Total":"1742.56",
                  "From Date":"02/29/16",
                  "To Date":"12/30/16",
                  "Unassigned":"345.90",
                  "2 2016":"1742.56"
               },
               {
                  "Number":"A0003-B0003-C0003-D0003-E0003",
                  "Name":"Row-00003",
                  "Profile":"S Curve",
                  "Total":"800.56",
                  "From Date":"01/01/17",
                  "To Date":"12/30/17",
                  "Unassigned":"300.90",
```

```
            "2 2016":"800.56"
         }
      ]
   },
   {
      "name":"Forecast",
      "Type":"Forecast",
      "alert_message":"",
      "currency":"Indian Rupee (INR)",
      "Total":"800.56",
      "From Date":"02/02/19",
      "To Date":"04/12/19",
      "Total":"1742.56"
      "Unassigned":"7443.20",
      "2 2016":"689.53"
      "details":[
         {
            "Number":"A0009-B0009-C0009-D0009-E00011",
            "Name":"Row-00011",
            "Profile":"S Curve",
            "Total":"800.56",
            "From Date":"02/02/19",
            "To Date":"04/12/19",
            "Unassigned":"7443.20",
            "2 2016":"689.53"
         },
         {
            "Number":"A0007-B0007-C0007-D0007-E0007",
            "Name":"Row-00007",
            "Profile":"S Curve",
            "Total":"800.56",
            "From Date":"05/05/19",
            "To Date":"12/30/19",
            "Unassigned":"280.43",
```

```
                    "2 2016":"3321.56"
                 }
              ]
           },
           {
              "alert_message":"",
              "Name":"Original",
              "Currency":"Canada Dollar (CAD)",
              "Type":"Portfolio Budget",
              "From Date":"",
              "To Date":"",
              "Total":"0.00",
              "Unassigned":"0.00",
              "1 2017":"100.00",
              "2 2017":"",

           }
        ]
     }
  ],
  "message":[
     "success"
  ],
  "status":"200"
}
```


Get Summary Cashflow Data

GET /ws/rest/service/v1/cashflow/summary/data/{project_number}

Purpose:

Get Summary Cashflow curve data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number (Required): Specify the project/shell number.

Url Parameter

filter (Required): Filter condition to be used when retrieving the data. It is a JSON object with the below information:

"name"(Required): Specify the cashflow name to get the data

"curves"(Optional): Specify the curve name for which the data is to be retrieved. Invalid curve name will be ignored. If no value is specified then all the curves data will be sent in the response.

Sample filter parameter

```
filter={  "name":"cashflow 1",   "curves":[  "Baseline", "Forecast"  ] }
```

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK , if success

For Partial failure, relevant message will be displayed ..

Sample output request

```
{
   "data":[
      {
         "name":"cashflow 1",
         "alert_message":[
            {
               "projectname":"Shell-Weekly",
               "name":"CF-New",
               "curvename":"Actuals",
               "errormessage":"Curve timescale is yearly"
            },
            {
               "projectname":"SUNEEL-RETEST-YEARLY",
               "name":"CashFlow-Suneel-Retest-Yearly",
               "curvename":"Actuals",
               "errormessage":"Curve timescale is yearly"
            }
         ],
         "curves":[
```

```
{
   "name":"Baseline",
   "alert_message":"",
   "currency":"Indian Rupee (INR)",
   "Type":"Baseline",
   "Total":"1742.56",
   "From Date":"02/29/16",
   "To Date":"12/30/16",
   "Unassigned":"345.90",
   "2 2016":"1742.56"
   "details":[
      {
         "Number":"A0001-B0001-C0001-D0001-E0001",
         "Name":"Row-00001",
         "Profile":"Front Load",
         "Total":"1742.56",
         "From Date":"02/29/16",
         "To Date":"12/30/16",
         "Unassigned":"345.90",
         "2 2016":"1742.56"
      },
      {
         "Number":"A0003-B0003-C0003-D0003-E0003",
         "Name":"Row-00003",
         "Profile":"S Curve",
         "Total":"800.56",
         "From Date":"01/01/17",
         "To Date":"12/30/17",
         "Unassigned":"300.90",
         "2 2016":"800.56"
      }
   ]
},
{
```

```
    "name":"Forecast",
     "Type":"Forecast",
    "alert_message":"",
    "currency":"Indian Rupee (INR)",
    "Total":"800.56",
    "From Date":"05/05/19",
    "To Date":"12/30/19",
    "Unassigned":"280.43",
    "2 2016":"3321.56"
    "details":[
       {
          "Number":"A0009-B0009-C0009-D0009-E00011",
          "Name":"Row-00011",
          "Profile":"S Curve",
          "Total":"800.56",
          "From Date":"02/02/19",
          "To Date":"04/12/19",
          "Unassigned":"7443.20",
          "2 2016":"689.53"
       },
       {
          "Number":"A0007-B0007-C0007-D0007-E0007",
          "Name":"Row-00007",
          "Profile":"S Curve",
          "Total":"800.56",
          "From Date":"05/05/19",
          "To Date":"12/30/19",
          "Unassigned":"280.43",
          "2 2016":"3321.56"
       }
    ]
},
{
    "alert_message":"",
```

```
                "Name":"Original",
                "Currency":"Canada Dollar (CAD)",
                "Type":"Portfolio Budget",
                "From Date":"",
                "To Date":"",
                "Total":"0.00",
                "Unassigned":"0.00",
                "1 2017":"100.00",
                "2 2017":"",

            }
         ]
      }
   ],
   "message":[
      "success"
   ],
   "status":"200"
}
```


Get Rollup Status

GET /ws/rest/service/v1/cashflow/rollup/status/{project_number}

Purpose:

Get Cashflow Rollup Status

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number to get the rollup status of project cashflows

url parameter(Optional)

filter=

{

"names":["Cash Flow 1" ,"Cash Flow 2"]

}

Filter can be used to filter on name to get the cash flow rollup status. If cashflow does not exist with the given name then that name will be ignored. Cashflows of other names will be returned.

If project number is not specified in the URL then error message will be returned

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

Get Rollup Status Ouptut JSON

```
{
    "data": [
        {
            "name": "Cash Flow 1",
            "id": "21",
            "status": "Active"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```

Get Rollup Status - For Template Cashflows

GET /ws/rest/service/v1/cashflow/template/rollup/status/{template_number}

Purpose:

Get Rollup Status of template cashflows

Input:

All parameters should be URL encoded.

Path Parameter

template_number(Optional): Specify the template number to get the rollup status of template cashflows otherwise to get the rollup status of standard template cashflows

url parameter(Optional)

filter=

{

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

Filter can be used to filter on name to get the cashflow rollup status. If cashflow does not exist with the given name then that name will be ignored. Cashflows of other names will be returned.

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

Get Rollup Status Ouptut JSON

```
{
    "data": [
        {
            "name": "Cash Flow 1",
            "id": "21",
            "status": "Active"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```

Update Rollup Status

PUT /ws/rest/service/v1/cashflow/rollup/status/{project_number}

Purpose:

Update Cashflow Rollup status

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number to update the rollup status of project cashflow

Both input & output in JSON format in the body

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Update Rollup Status Input JSON

```
{
    "data" : [
                {"name": "Cash Flow 1", "rollup_status": "Inactive"},
                {"name": "Cash Flow 2", "rollup_status": "Open"},
                {"name": "Cash Flow 3", "rollup_status": "Active"}
    ]
}
```

Update Rollup Status Ouptut JSON

```
{
    "data": [
        {
            "rollup_status": "Active",
            "name": "Cash Flow 1",
            "id": "61"
        }
    ],
    "message": [
        {
            "name": "Cash Flow 1",
            "message": "success",
            "status": 200
        },
        {
            "name": "Cash Flow 2",
            "message": "Status field value is not correct.",
            "status": 663
        },
        {
```

```
            "name": "Cash Flow 3",
            "message": "Cash Flow do not exist",
            "status": 1305
        }
    ],
    "status": 3000
}
```

Update Rollup Status - For Template Cashflows

PUT /ws/rest/service/v1/cashflow/template/rollup/status/{template_number}

Purpose:

Update Rollup status of template cashflows

Input:

All parameters should be URL encoded.

Path Parameter

template_number(Optional): Specify the template number to update the rollup status of template cashflows otherwise to update the rollup status of standard template cashflows

Both input & output in JSON format in the body

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Update Rollup Status Input JSON

```
{
    "data" : [
                {"name": "Cash Flow 1", "rollup_status": "Inactive"},
                {"name": "Cash Flow 2", "rollup_status": "Open"},
                {"name": "Cash Flow 3", "rollup_status": "Active"}
    ]
}
```

Update Rollup Status Ouptut JSON

```
{
    "data": [
```

```
        {
            "rollup_status": "Active",
            "name": "Cash Flow 1",
            "id": "61"
        }
    ],
    "message": [
        {
            "name": "Cash Flow 1",
            "message": "success",
            "status": 200
        },
        {
            "name": "Cash Flow 2",
            "message": "Status field value is not correct.",
            "status": 663
        },
        {
            "name": "Cash Flow 3",
            "message": "Cash Flow do not exist",
            "status": 1305
        }
    ],
    "status": 3000
}
```


Refresh Cashflow

PUT /ws/rest/service/v1/cashflow/refresh/{project_number}

Purpose:

Refresh Cashflow Properties

Input:

All parameters should be URL encoded.

Path Parameter

project_number:Required

Path Parameter

Both input & output in JSON format in the body

{

"names": ["CBS CF", cash_flow_2"]

}

Supported options properties

"names": ["CBS CF", "cash_flow_2"] for one or more values

Or "names": "CBS CF" for only one value

"rollup_status"<<Active><Inactive>> (or "status" in case)

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_levelvalue provided is other than above mentioned values, invalid options message will be thrown.

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK , if success

For Partial failure, relevant message will be displayed.

Refresh Cashflow Properties Sample Response

1) response when success:

{

"data": "",

"message": ["http://slc12knf.us.oracle.com:912<server url>/ws/rest/service/v1/cashflow/jobs/1503681423559"],

"status": 202

}

2) response when partial success:

{

"invalid_names": "[Summary1, Curve2]",

"data": [],

"message": [

"http://blr00bti.in.oracle.com:3883/idchyd<server url>/ws/rest/service/v1/cashflow/jobs/1556455198417"

],

"status": 3000

}

Get Cashflow Refresh Job Status

GET /ws/rest/service/v1/cashflow/jobs/{job_id}

Purpose:

To retrieve Cashflow refresh job status

Input:

All parameters should be URL encoded.

Path Parameter

job_id: Required

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK , if success

For Partial failure, relevant message will be displayed.

Refresh Cashflow Job status Sample Response

1) on success.

```
{
    "data":"",
    "job_status":"Finished",
    "status_code":200,
    "message":"OK"
}
```

2) when job is in progress.

```
{
    "data":"",
    "job_status":"In progress",
    "status_code":200,
    "message":"OK"
}
```

3) when job has failed.

```
{
    "data":"Failure detail",
    "job_status":"Finished with errors",
    "status_code":1006,
    "message":"Job processing failed with errors."
}
```

Get Cashflow Properties

GET /ws/rest/service/v1/cashflow/{project_number}

Purpose:

Get Cashflow properties

Input:

All parameters should be URL encoded.

Path Parameter

project_number: optional, if not provided then company cash flow will be fetched.

input query parameter -

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

Supported options in filter:

"names": "CBS CF"

"curve_name": << Name of The Curve>>

"rollup_status"<<Active><Inactive>>

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_level value provided is other than above mentioned values, invalid options message will be thrown.

"include_curves":<< Yes/ No>> <!-- default is Yes if not provided→>

If include_curves value provided is other than above mentioned values, invalid options message will be thrown.

Note: if no filter options provided service will return all cashflow curves from the project.

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK , if success

For Partial failure, relevant message will be displayed.

Project or Shell


```
{
    "data": [
        {
            "name": "Project_CF",
            "description": "test webservice get Cashflow Properties",
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 8,
            "snapshot": {
                "snapshot_day_of": 1,
                "cutoff_spends": {
                    "cutoff_week_num": "First",
                    "cutoff_week_day": "Sunday",
                    "inc_spends_opt": "next_month"
                }
            },
            "curves": [
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
                        "type": "auto_profile",
                        "distribution_profile": "Front Loaded"
                    },
                    "cost": {
                        "type": "auto_distribution"
```

```
            },
            "schedule": {
                "type": "schedule_sheet",
                "sheet_name": "sheet3",
                "sch_start_source": "Estimated Finish Date",
                "sch_end_source": "Estimated Start Date"
            }
        },
        {
            "name": "Forecast",
            "type": "Forecast",
            "distribution": {
                "type": "manual"
            },
            "cost": {
                "type": "manual"
            },
            "schedule": {
                "type": "schedule_sheet",
                "sheet_name": "Sheet1",
                "sch_start_source": "AP_Date6",
                "sch_end_source": "AP_Date only 1"
            },
            "forecast_options": {
                "begin_end_of_curve": true,
                "fc_curve_name": "Actuals",
                "current_actual_cut_off_date": true,
                "edit_until_replace_by_actual": true
            }
        },
        {
            "name": "Actuals",
            "type": "Spends",
            "cost": {
```

```
                        "cost_sheet_column": "Purchase Orders
(Pending)",
                        "type": "cost_sheet"
                    }
                },
                {
                    "name": "Original Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Approved Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "RADO",
                        "distribut_amount_from": "approved"
                    }
                },
                {
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Afghani (AFN)",
                        "distribut_amount_from": "shared"
                    }
                },
                {
                    "name": "custom_D",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
```

```
                    "source_curve_name": "Baseline",
                    "derived_curve_currency": "Afghani (AFN)",
                    "exchange_rate": "Float",
                    "float_rate_as": "end_period"
                }
            }
        ],
        "filters": {
            "filter_option": "all"
        },
        "schedule":{
            "enable_refresh" : true,
            "frequency" : "Quarterly",
            "quarterday" : 4,
            "startdate": "2019/03/30",
            "noenddate": false,
            "enddate"    : "2020/03/30"
        },
        "summary_curves":[
            {
                "name":"curve_1",
                "summary_type" : "System Defined",
                "status" : "Active"
            },
            {
                "name":"curve_2",
                "summary_type" : "System Defined",
                "status" : "Active"
            }
        ]
    }
],
"message": [
    "success"
```

```
    ],
    "status": 200
}
```


CBS


```
{
    "data": [
        {
            "name": "CBS_CF",
            "description": "CBS  Cashflow",
            "rollup_status": "Active",
            "detail_level": "CBS",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Year",
            "period_format": "YYYY",
            "decimal_places": 3,
            "snapshot": {
                "snapshot_week_num": 1,
                "snapshot_week_day": "1"
            },
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "cost_sheet_column": "Purchase Orders
(Pending)",
                        "type": "cost_sheet"
                    }
                },
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
```

```
                        "type": "auto_profile",
                        "cbs": [
                            {
                                "cbs_code": "s1-a 1-b 1-a123~~s3-a 2-b
1-a123",
                                "bitemid": 13,
                                "item": "A123",
                                "default_profile": "Front Loaded"
                            },
                            {
                                "cbs_code": "s1-a 1-b 1-a123~~s2-a 2-b
1-a123",
                                "bitemid": 12,
                                "item": "A123",
                                "default_profile": null
                            }
                        ]
                    },
                    "cost": {
                        "type": "auto_distribution"
                    },
                    "schedule": {
                        "type": "schedule_sheet",
                        "sheet_name": "Sheet1",
                        "sch_start_source": "Finish date",
                        "sch_end_source": "Late Start date"
                    }
                },
                {
                    "name": "Forecast",
                    "type": "Forecast",
                    "distribution": {
                        "type": "schedule_sheet",
                        "sch_sheet_name": "sheet2"
```

```
                    },
                    "cost": {
                        "type": "manual"
                    },
                    "schedule": {
                        "type": "manual"
                    },
                    "forecast_options": {
                        "begin_end_of_curve": true,
                        "fc_curve_name": "Actuals",
                        "current_actual_cut_off_date": true,
                        "fc_dist": "weighted_avg_over_next",
                        "periods": 1
                    }
                },
                {
                    "name": "Protfolio Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Costa Rica Colon (CRC)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "custom_D",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Float",
                        "float_rate_as": "weighted_period"
                    }
```

```
},
{
    "name": "Approved Budget",
    "type": "Portfolio Budget",
    "cost": {
        "currency_code": "Bhutan Ngultrum (BTN)",
        "distribut_amount_from": "approved"
    }
},
{
    "name": "Original Budget",
    "type": "Portfolio Budget",
    "cost": {
        "currency_code": "RADO",
        "distribut_amount_from": "original"
    }
},
{
    "name": "Shared Budget",
    "type": "Portfolio Budget",
    "cost": {
        "currency_code": "RADO",
        "distribut_amount_from": "shared"
    }
},
{
    "name": "Derive",
    "type": "Derived",
    "cost": {
        "base_currency": "United States Dollar (USD)",
        "source_curve_name": "Forecast",
        "derived_curve_currency": "Dirham (AED)",
        "exchange_rate": "Float",
        "float_rate_as": "beginning_period"
```

```
            }
        },
        {
            "name": "Forecast 2",
            "type": "Forecast",
            "distribution": {
                "type": "manual"
            },
            "cost": {
                "type": "manual"
            },
            "schedule": {
                "type": "p6_sheet",
                "p6_datasource": "Original Baseline",
                "sch_dates_type": "At Completion"
            },
            "forecast_options": {}
        }
    ],
    "filters": {
        "filter_option": "all"
    },
    "schedule":{
        "enable_refresh" : true,
        "frequency" : "Quarterly",
        "quarterday" : 4,
        "startdate": "2019/03/30",
        "noenddate": false,
        "enddate"    : "2020/03/30"
    },
    "summary_curves":[
        {
            "name":"curve_1",
            "summary_type" : "System Defined",
```

```
                    "status" : "Active"
                },
                {
                    "name":"curve_2",
                    "summary_type" : "System Defined",
                    "status" : "Active"
                }
            ]
         }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Summary CBS


```
{
    "data": [
        {
            "name": "SummaryCBS_CF",
            "description": "SummaryCBS_CF for Cashflow Properties",
            "rollup_status": "Active",
            "detail_level": "Summary CBS",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 3,
            "snapshot": {
                "snapshot_week_num": 1,
                "snapshot_week_day": "2",
                "cutoff_spends": {
                    "cutoff_week_num": 2,
```

```
                    "cutoff_week_day": "6",
                    "inc_spends_opt": "next_month"
                }
            },
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "cost_sheet_column": "Purchase Orders
(Pending)",
                        "type": "cost_sheet"
                    }
                },
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
                        "type": "auto_profile"
                    },
                    "cost": {
                        "type": "cost_sheet",
                        "cost_sheet_column": "Purchase Orders (Pending)"
                    },
                    "schedule": {
                        "type": "schedule_sheet",
                        "sheet_name": "Sheet1",
                        "sch_start_source": "Start date",
                        "sch_end_source": "Finish date"
                    }
                },
                {
                    "name": "Forecast",
                    "type": "Forecast",
```

```
            "distribution": {
                "type": "manual"
            },
            "cost": {
                "type": "manual"
            },
            "schedule": {
                "type": "manual"
            },
            "forecast_options": {}
        },
        {
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "test",
                "distribut_amount_from": "approved"
            }
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Yen (JPY)",
                "distribut_amount_from": "original"
            }
        },
        {
            "name": "Shared Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Algeria Dinar (DZD)",
                "distribut_amount_from": "shared"
            }
```

```
                },
                {
                    "name": "Derive",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Baseline",
                        "derived_curve_currency": "Bhutan Ngultrum
(BTN)",
                        "exchange_rate": "Float",
                        "float_rate_as": "beginning_period"
                    }
                }
            ],
            "filters": {
                "filter_option": "all"
            },
            "schedule":{
                "enable_refresh" : true,
                "frequency" : "Quarterly",
                "quarterday" : 4,
                "startdate": "2019/03/30",
                "noenddate": false,
                "enddate"    : "2020/03/30"
            },
            "summary_curves":[
                {
                    "name":"curve_1",
                    "summary_type" : "System Defined",
                    "status" : "Active"
                },
                {
                    "name":"curve_2",
                    "summary_type" : "System Defined",
```

```
                    "status" : "Active"
                }
            ]
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Commitment Type


```
{
    "data": [
        {
            "name": "Commitment_CF",
            "description": "commit",
            "rollup_status": "Active",
            "detail_level": "Commitment",
            "bp_name": "Purchase Orders",
            "reference_elements": "record_no:Record No.",
            "base_commit_record": "PO-0001",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cashflow_granularity": "by_lineitem",
                    "cost": {
                        "spends_bp": [{"bp_name" :"A_Commit_01",
"status":"Approved"}]
```

```
                    }
                },
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "cashflow_granularity": "individual_record",
                    "distribution": {
                        "type": "auto_profile",
                        "commit_profile": "business_process",
                        "bp_dates_profile": "{ Purchase Orders:Creation
Date:Creation Date:Back Loaded }{ PO Amendments:Creation Date:Creation
Date:Linear }"},
"bp_dates_profile": [
                     {
                       "bp_name": "Purchase Orders",
                        "todate": "uuu_creation_date ",
                        "profile": "Front Loaded",
                        "fromdate": "ugenP6PlannedStartDOP"
                      }
                ],
                    "commit_change_processes": "{PO
Amendments:Approved}{PO Amendments:Canceled}{PO Amendments:Rejected}"
"commit_change_processes": [
                        {
                            "bp_name": "PO Amendments",
                            "status": "Approved"
                        },
                        {
                           "bp_name": "PO Amendments",
                            "status": "Canceled"
                        },
                        {
                            "bp_name": "PO Amendments",
                            "status": "Pending"
```

```
            },
            {
                "bp_name": "PO Amendments",
                "status": "Rejected"
            }
        ]
    },
    {
        "name": "Forecast",
        "type": "Forecast",
        "cashflow_granularity": "by_lineitem",
        "distribution": {
            "type": "manual"
        },
        "commit_change_processes": "",
        "forecast_options": {}
    },
    {
        "name": "Approved Budget",
        "type": "Portfolio Budget",
        "cost": {
            "currency_code": "Costa Rica Colon (CRC)",
            "distribut_amount_from": "approved"
        }
    },
    {
        "name": "Shared Budget",
        "type": "Portfolio Budget",
        "cost": {
            "currency_code": "Dirham (AED)",
            "distribut_amount_from": "shared"
        }
    },
    {
```

```
                    "name": "Original Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "Afghani (AFN)",
                        "distribut_amount_from": "original"
                    }
                },
                {
                    "name": "Derive",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Baseline",
                        "derived_curve_currency": "Costa Rica Colon
(CRC)",
                        "exchange_rate": "Float",
                        "float_rate_as": "weighted_period"
                    }
                }
            ],
            "schedule":{
                "enable_refresh" : true,
                "frequency" : "Quarterly",
                "quarterday" : 4,
                "startdate": "2019/03/30 00:00",
                "noenddate": false,
                "enddate": "2020/03/31 00:00"
            },
            "summary_curves":[
                {
                    "name":"curve_1",
                    "summary_type" : "System Defined",
                    "status" : "Active"
                },
```

```
                {
                    "name":"curve_2",
                    "summary_type" : "System Defined",
                    "status" : "Active"
                }
            ]
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
{
    "data": [
        {
            "id": 670,
            "name": "P-0006",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Commitment",
            "bp_name": "Purchase Orders",
            "reference_elements": "record_no:record_no",
            "base_commit_record": "PO-0005",
            "period_type": "Financial Periods",
            "period_name": "FP_02",
            "period_by": "Custom",
            "decimal_places": 2,
            "schedule": {
                "enable_refresh": true,
                "frequency": "Quarterly",
                "quarterday": 4,
                "startdate": "06/24/2019",
                "noenddate": false,
```

```
        "enddate": "06/24/2020"
    },
    "curves": [
        {
            "id": 930,
            "name": "Forecast",
            "type": "Forecast",
            "cashflow_granularity": "by_lineitem",
            "distribution": {
                "type": "manual"
            },
            "commit_change_processes": [],
            "forecast_options": {}
        },
        {
            "id": 931,
            "name": "Baseline",
            "type": "Baseline",
            "cashflow_granularity": "individual_record",
            "distribution": {
                "type": "auto_profile",
                "commit_profile": "business_process",
                "bp_dates_profile": [
                    {
                        "bp_name": "Purchase Orders",
                        "todate": "uuu_creation_date",
                        "profile": "Back Loaded",
                        "fromdate": "ugenP6PlannedStartDOP"
                    },
                    {
                        "bp_name": "PO Amendments",
                        "todate": "ugenP6PlannedFinishDOP",
                        "profile": "Linear",
                        "fromdate": "due_date"
```

```
                }
            ]
        },
        "commit_change_processes": [
            {
                "bp_name": "PO Amendments",
                "status": "Approved"
            },
            {
                "bp_name": "PO Amendments",
                "status": "Pending"
            }
        ]
    },
    {
        "id": 957,
        "name": "Actuals",
        "type": "Spends",
        "cashflow_granularity": "by_lineitem",
        "cost": {
            "spends_bp": [
                {
                    "bp_name": "Invoices",
                    "status": "Approved"
                },
                {
                    "bp_name": "Invoices",
                    "status": "Pending"
                }
            ]
        }
    },
    {
        "id": 958,
```

```
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Belize Dollar (BZD)",
                "distribut_amount_from": "approved"
            }
        },
        {
            "id": 959,
            "name": "Shared Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Indian Rupee (INR)",
                "distribut_amount_from": "shared"
            }
        },
        {
            "id": 960,
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "Malta Lira (MTL)",
                "distribut_amount_from": "original"
            }
        },
        {
            "id": 961,
            "name": "Derived",
            "type": "Derived",
            "cost": {
                "base_currency": "United States Dollar (USD)",
                "source_curve_name": "Baseline",
                "derived_curve_currency": "Belize Dollar (BZD)",
                "exchange_rate": "Float",
```

```
                        "float_rate_as": "weighted_period"
                    }
                }
            ],
            "summary": [
                {
                    "name": "Cash Flow Summary Curve",
                    "summary_type": "System_Defined"
                }
            ]
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```

All possible Schedule values

```
"schedule":{
    "enable_refresh" : false
}
"schedule":{
    "enable_refresh" : true,
    "frequency" : "Daily",
    "startdate": "2019/03/30 00:00",
    "noenddate": true,
    "enddate": null
}
"schedule":{
    "enable_refresh" : true,
    "frequency" : "Weekly",
    "weekday"    : "Thursday",
    "startdate": "2019/03/30 00:00",
```

```
    "noenddate": false,
    "enddate": "2020/03/31 00:00"
}
"schedule":{
    "enable_refresh" : true,
    "frequency" : "Monthly",
    "bymonthday": true,
    "monthday": 11,
    "startdate": "2019/03/30 00:00",
    "noenddate": false,
    "enddate": "2020/03/31 00:00"
}
"schedule":{
    "enable_refresh" : true,
    "frequency" : "Monthly",
    "bymonthday": false,
    "monthweeknum"    : 2
    "monthweekday": "Thursday",
    "startdate": "2019/03/30 00:00",
    "noenddate": false,
    "enddate": "2020/03/31 00:00"
}
"schedule":{
    "enable_refresh" : true,
    "frequency" : "Quarterly",
    "quarterday" : 4,
    "startdate": "2019/03/30 00:00",
    "noenddate": false,
    "enddate": "2020/03/31 00:00"
}
```

Company or Program Level

```
{
    "data": [
```

```
        {
            "name": "Company cashflow- roll up curves",
            "description": null,
            "status": "Active",
            "detail_level": "Company", ????????????? do we need to
include for company/program level as they are not detail level curves
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "filters": {
                "filter_option": "all"
            },

            "data_source": [
                {
                    "name": "Baseline",
                    "curve_type": "Baseline"
                },
                {
                    "name": "Forecast",
                    "curve_type": "Forecast"
                },
                {
                    "name": "Actuals",
                    "curve_type": "Spends"
                }
            ]
        }
    ],
    "message": [
        "success"
    ],
```

```
    "status": 200
}
```

Get Summary Cashflow Properties

GET /ws/rest/service/v1/cashflow/summary/{project_number}

Purpose:

Get Summary Cashflow properties

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Mandatory): Specify the project/Shell number to get project cashflow properties

input query parameter -

filter=

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

Filter can be used to filter on name to get the summary cash flow properties. If summary cashflow does not exist with the given name then that name will be ignored.

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 , for success

Get Summary Cashflow Properties

```
{
    "data": [
        {
            "curves": [
                "Cash Flow 1",
                "Cash flow 2"
            ],
            "name": "Sum Cash Flow 1",
            "description": "This summary curve is created from REST API",
            "summary_type": "User_Defined",
```

```
            "status": "Active"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```

Get Cashflow Permissions

GET /ws/rest/service/v1/cashflow/permission/{project_number}

Purpose:

Fetch Cash flow Permissions

The input JSON shall provide various options to be considered for fetching cashflow permissions.

POST body is a JSON

Note: POST call has input & output both as JSON in the body

input query parameter

filter =

```
{
    "names" : [ "cash flow 1", "cash flow 2" ]
 }
```

Input:

All parameters should be URL encoded.

project_number: Specify the Project number in which the cash flow exists. If not provided, Company workspace will be considered.

In request, "names" is optional.

In the request body names can be provided in below two formats

1. "names": ["cash flow 1", "cash flow 2", "cash flow 3"] For one or more cash flows.
2. "names": "cash flow 1" For only one cash flow.

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

Get Cash flow Permission Sample Response

```
{
    "data": {
        "cash flow 1": {
           "permissions": [
                {
                    "login_name": "coadmin",
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "U"
                },
                {
                    "full_name": "Company Group 1",
                    "group_name": "Company Group 1",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "CG"
                }
            ]
        },
        "Cash flow 2": {
            "permissions": [
                {
                    "login_name": "firstname",
                    "full_name": "Firstname Lastname",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
```

```
                    "modify_permission": "1"
                },
                "type": "U"
            }
        ]
    }
},
"message": [
    "success"
],
"status": 200
}
```

Update or Modify Cashflow Permission

PUT /ws/rest/service/v1/cashflow/permission/{project_number}

Purpose:

Update permission (in user mode) of a specific cashflow in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the cashflow exists, if not provided then cashflows are considered to be fetched from Company Level.

POST body is a JSON

Note: POST call has input & output both as JSON in the body

Input JSON

```
{
    "data" :
    {
        "names": ["cash flow 1", "cash flow 2", "cash flow 3"] ,
        "permissions": [
                {
                    "login_name": "coadmin",
                    "type": "U",
```

```
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "modify_permission": "0",
                        "view": "1"
                    }
                },
                {
                    "full_name": "Project Manager 1",
                    "group_name": "Project Manager 1",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "PG"
                }
            ]
        }
    }
```

In request, "names" is mandatory .

In the request body names can be provided in below two formats

1. "names": ["cash flow 1", "cash flow 2", "cash flow 3"] For one or more cash flows.
2. "names": "cash flow 1" For only one cash flow.

If names not provided, then error response will be provided to user to provide cash flows.

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

Update or Modify Cash flow Permission Sample Response

1) Response for success case:

```
{
    "data": {
        "cash flow 1": {
            "permissions": [
                {
                    "login_name": "donna",
```

```
            "full_name": "Donna Pinciotti",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "1"
            },
            "type": "U"
        },
        {
            "login_name": "coadmin",
            "full_name": "Company Administrator",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "0"
            },
            "type": "U"
        },
        {
            "full_name": "Project Managers",
            "group_name": "Project Managers",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "0"
            },
            "type": "PG"
        }
    ],
    "updated_users": [
        "coadmin"
    ],
    "updated_groups": [
        "Project Managers"
```

```
        ]
    },
    "cash flow 2": {
        "permissions": [
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "U"
            }
        ],
        "updated_users": [],
        "updated_groups": []
    },
    "cash flow 3": {
        "permissions": [
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "0"
                },
                "type": "U"
            },
            {
                "login_name": "donna",
                "full_name": "Donna Pinciotti",
                "permission": {
```

```
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                },
                {
                    "full_name": "Project Managers",
                    "group_name": "Project Managers",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "PG"
                }
            ],
            "updated_users": [
                "coadmin"
            ],
            "updated_groups": [
                "Project Managers"
            ]
        }
    },
    "message": [
        "success"
    ],
    "status": 200
}
```


2: Response for partial success case:


```
{
    "data": {
```

```
"cash flow 1": {
    "permissions": [
        {
            "login_name": "donna",
            "full_name": "Donna Pinciotti",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "1"
            },
            "type": "U"
        },
        {
            "login_name": "coadmin",
            "full_name": "Company Administrator",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "0"
            },
            "type": "U"
        },
        {
            "full_name": "Project Managers",
            "group_name": "Project Managers",
            "permission": {
                "edit_data": "1",
                "view": "1",
                "modify_permission": "0"
            },
            "type": "PG"
        }
    ],
    "updated_users": [
```

```
                "coadmin"
            ],
            "updated_groups": [
                "Project Managers"
            ]
        }
    },
    "message": [
        {
            "name": "cash flow 2",
            "status": "update permission error message"
        },
        {
            "name": "cash flow 3",
            "status": "update permission error message"
        }
    ],
    "status": 3000
}
```


Create (Add User or Group) Cashflow Permission

POST /ws/rest/service/v1/cashflow/permission/{project_number}

Purpose:

Add User or Group to Cashflow permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the Cashflow exists, if not provided then Cashflows are considered to be fetched from Company Level.

POST body is a JSON

Note: POST call has input & output both as JSON in the body

Input JSON

```
{
    "data" :
    {
        "names": ["cash flow 1"] ,
        "permissions": [
                {
                    "login_name": "PU1",
                    "type": "U",
                    "full_name": "P1 user",
                    "permission": {
                        "edit_data": "1",
                        "modify_permission": "1",
                        "view": "1"
                    }
                }
            ]
    }
}
```

Here "cash_flow" are mandatory in options.

In the request body names can be provided in below two formats

1. "names": ["cash flow 1", "cash flow 2", "cash flow 3"] For one or more cash flows.
2. "names": "cash flow 1" For only one cash flow.

If names not provided, then error response will be provided to user to provide cash flows .

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

Create (Add User/Group) Cash flow Permission Sample Response

1) Response for success case:

```
{
    "data": {
        "cash flow 1": {
            "added_users": [PU1],
            "permissions": [
                {
                    "login_name": "PU1",
```

```
                "full_name": "PU1 P",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "U"
            },
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "U"
            },
            {
                "full_name": "Space Planners",
                "group_name": "Space Planners",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "CG"
            }
        ],
        "added_groups": []
    }
},
"message": [
    "success"
```

```
    ],
    "status": 200
}
```


2) Response for partial success case:


```
{
    "data": {
        "cash flow 1": {
            "permissions": [
                {
                    "login_name": "donna",
                    "full_name": "Donna Pinciotti",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                },
                {
                    "full_name": "Project Managers",
                    "group_name": "Project Managers",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "PG"
                }
            ],
            "added_users": [
                "donna"
            ],
            "added_groups": [
```

```
                "Project Managers"
            ]
        }
    },
    "message": [
        {
            "name": "cash flow 2",
            "status": "add permission error message"
        },
        {
            "name": "cash flow 3",
            "status": "add permission error message"
        }
    ],
    "status": 3000
}
```

Delete (Remove User or Group) Cashflow Permission

DELETE /ws/rest/service/v1/cashflow/permission/{project_number}

Purpose:

Add User/Group to Cashflow permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the Cashflow exists, if not provided then Cashflows are considered to be fetched from Company Level.

POST body is a JSON

Note: POST call has input & output both as JSON in the body

Input JSON

```
{
    "data" :
    {
        "names": ["cash flow 1"],
```

```
            "user_names" : ["coadmin","donna","PU1"],
            "group_names" : [ "group1", "group2" ]
        }
    }
```

Here "cash_flow" are mandatory in options.

In the request body names can be provided in below two formats

1. "names": ["cash flow 1", "cash flow 2", "cash flow 3"] For one or more cash flows.
2. "names": "cash flow 1" For only one cash flow.

If names not provided, then error response will be provided to user to provide cash flows .

Output:

JSON object containing 'status', 'data', 'message'

message will be present if status is not 200 otherwise it will be "success".

Delete (Remove User/Group) Cash flow Permission Sample Response

1) Response for success case:

```
{
    "data": {
        "cash flow 1": {
            "deleted_groups": [],
            "deleted_users": [
                "PU1 P"
            ]
        }
    },
    "message": [
        "success"
    ],
    "status": 200
}
```

2) Response for partial success case:

```
{
    "data": {
        "cash flow 1": {
            "deleted_groups": [],
```

```
            "deleted_users": [
                "PU1 P"
            ]
        }
    },
    "message": [
        {
            "name": "cash flow 2",
            "status": "delete permission error message"
        },
        {
            "name": "cash flow 3",
            "status": "delete permission error message"
        }
    ],
    "status": 3000
}
```


Update Cashflow

PUT /ws/rest/service/v1/cashflow/{project_number}

Purpose:

Update cashflow properties data. Can be used to add new curves to cash flow.

Input:

All parameters should be URL encoded.

Specify only the data that needs to be updated. Use the response from get cashflow properties service "GET /ws/rest/service/v1/cashflow/{project_number}" to create the update request.

Cashflow curves will be updated based on the cashflow id.

If the attribute is not provided then the value will not be updated.

If the attribute is provided with value then validation will be performed and the value will be updated.

When the attribute is provided and the value is empty, then the value will be deleted.

To delete a profile from profile list of CBS or Summary CBS , provide empty default profile, as below.

Irrelevant attributes specified in the request for create/update are ignored.

Deletion of Profile

```
"distribution": {
                    "type": "auto_profile",
                    "cbs": [
                            {
                                 "cbs_code": "t30001~~t30002",
                                 "bitemid": 242,
                                 "item": "T30002",
                                 "default_profile": ""
                             }
                        ]
                   }
```

To delete a change commit BP list, specify "action" attribute with value as "delete", as given below:

Deletion of Commit BP

```
"commit_change_processes": [
                        {
                            "bp_name": "PO Amendments",
                            "status": "Canceled",
                            "action":"delete"
                        }
                     ]
```

To update Distribution profiles in Commitment detail level Cashflow, All values of "that profile map" must be provided . Other maps (or rows in UI) of bp_date_profiles which are not subject to update, are optional (may or may not be in bp_date_profiles list).

```
"bp_dates_profile": [
                        {
                            "bp_name": "PO Amendments",
                            "todate": "ugenP6PlannedFinishDOP",
                            "profile": "Back Loaded",
                            "fromdate": "due_date"
                        }
                     ]
```

To delete Distribution profile in Commitment Detail level Cashflow, provide correct "bp_name" and "to_date", "fromdate" & "profile" as empty String, in "bp_dates_profiles.

```
"bp_dates_profile": [
                        {
                            "bp_name": "PO Amendments",
                            "todate": "",
                            "profile": "",
                            "fromdate": ""
                        }
                    ]
```

Note: POST call has input & output both as JSON in the body.

Update Cashflow Manual - Project or Shell

```
{
    "data":
        {   "id":12,
            "name": "project type cashflowupdated",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
           "snapshot": {
                "sn_week_num": "First",
                "sn_week_day": "Sunday",
                "cutoff_spends": {
                    "cutoff_week_num": "First",
                    "cutoff_week_day": "Sunday",
                    "inc_spends_opt": "same_month"
                }
            },
            "schedule": {
                "enable_refresh": false
```

```
            },

            "summary": [
                {
                    "name": "summary",
                    "summary_type": "User_Defined"
                }
            ],
            "curves": [
                {
                    "name": "Actuals",
                    "type": "Spends",
                    "cost": {
                        "type":"cost_sheet",
                        "cost_sheet_column": "152 Base Commit WF
(Approved)"
                    }
                },
                {
                    "name": "Cash flow3",
                    "type": "Derived",
                    "cost": {
                        "base_currency": "United States Dollar (USD)",
                        "source_curve_name": "Actuals",
                        "derived_curve_currency": "Australia Dollar
(AUD)",
                        "exchange_rate": "Peg",
                        "peg_rate_as": "Dynamic",
                        "peg_dynamic_ds": "Single Record BPs / 922_Simple
PNW/FSingle LL",
                        "peg_dynamic_de": "Currency amount 1",
                        "float_rate_as": "weighted_period"
                    }
                },
```

```
            {
                "name": "Today",
                "type": "Portfolio Budget",
                "cost": {
                    "currency_code": "Australia Dollar (AUD)",
                    "distribut_amount_from": "approved"
                }
            },
            {
                "name": "Forecast",
                "type": "Forecast",
                "distribution": {
                    "type": "auto_profile",
                    "distribution_profile": "Front Load"
                },
                "cost": {
                    "type": "cost_sheet",
                    "cost_sheet_column": "152 Base Commit WF (Open)"
                },
                "schedule": {
                    "type": "manual"
                },
                "forecast_options": {
                    "begin_end_of_curve": true,
                    "fc_curve_name": "Actuals",
                    "current_actual_cut_off_date": true,
                    "fc_dist": "weighted_avg_over_next",
                    "periods": 3
                }
            }
        ]
    }
```

```
}
```

For other detail_type sample output, please refer to create Cashflow call.

Output:

JSON object containing 'status', 'data', 'message'

a message will be present if the status is not 200 otherwise it will be "success".

Update Cashflow Data

PUT /ws/rest/service/v1/cashflow/data/{project_number}

Purpose:

This service will update the cashflow detail curve data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number.

PUT body is in JSON format.

"options": Specify if curve "setup" or "distribution" data has to be updated. If no data is provided then the default will be setup = "yes".

"setup": Optional, default is "yes". Specify "yes" if you need to update the curve setup data (From Date, To Date, Profile and Total), "no" if curve setup is not required to be updated.

"distribution": Optional, default is "no". Specify "yes" if you need to update the distribution data (distribution values of the Total amount entered in the date period), "no" if the distribution is not required to be updated.

If both setup and distribution are set to "yes" then only "setup" data will be updated.

if both setup and distribution are set to "no" then no data will be updated.

"data": Specify the curve data which needs to be updated. Use the response from the REST API, "/cashflow/data". Use the "details" part of the response.

The "From Date" and "To Date" should use integration user preference data format.

Sample Request

```
{
   "options":{
      "setup":"yes",
      "distribution":"no"
   },
   "data":[
      {
```

```
"Name":"cashflow 1",
"curves":[
   {
      "Type":"Forecast",
      "Name":"Forecast",
      "details":[
         {
            "Number":"TestProj-C",
            "Name":"TestProj-C-TestProj-C Name",
            "From Date":"05-31-2019 00:00:00",
            "To Date":"10-25-2019 00:00:00",
            "Profile":"",
            "Total":"230.00",
            "Unassigned":"-48.00",
            "5 2019":"450.00",
            "6 2019":"40.00",
            "7 2019":"10.00",
            "8 2019":"-44.00",
            "9 2019":"-44.00",
            "10 2019":"-44.00"
         }
      ]
   },
   {
      "Type":"Baseline",
      "Name":"Baseline",
      "details":[
         {
            "Number":"TestProj-C",
            "Name":"TestProj-C-TestProj-C Name",
            "From Date":"05-01-2019 00:00:00",
            "To Date":"10-25-2019 00:00:00",
            "Profile":"",
            "Total":"100.00",
```

```
                    "Unassigned":"90.00",
                    "5 2019":"5.00",
                    "6 2019":"0.00",
                    "7 2019":"0.00",
                    "8 2019":"0.00",
                    "9 2019":"10.00",
                    "10 2019":"0.00"
                }
            ]
        }
    ]
  }
]
}
```


Output:

JSON object containing 'status', 'data', 'message'

a message will be present if the status is not 200 otherwise it will be "success".

Update Data Response


```
{
    "data": [],
    "message": [
        {
            "curve_name": "Baseline",
            "status_code": 200,
"name": "cashflow 1"
            "message": "success"
        }
    ],
    "status": 200
}
```


Create Distribution Profiles

POST /ws/rest/service/v1/cashflow/profile

Purpose:

Create distribution profiles of company

Input:

Both input & output in JSON format in the body.

Create Distribution Profile input JSON


```
{
    "data": [
        {
            "name": "S Curve new",
            "status": "Active",
            "distribution": [
                "0.5",
                "0.5",
                "1.5",
                "1.5",
                "4.0",
                "4.0",
                "7.5",
                "7.5",
                "11.5",
                "11.5",
                "11.5",
                "11.5",
                "7.5",
                "7.5",
                "4.0",
                "4.0",
                "1.5",
                "1.5",
                "0.5",
                "0.5"
            ]
        },
```

```
        {
            "name": "Front Loaded",
            "status": "Active",
            "distribution": [
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "6.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5"
            ]
        }
    ]
}
```

Supported values for "status" is "Active" or "Inactive".

The "distribution" must be a list of at least 20 elements, the sum of them all must be 100. If the "distribution" is not provided, then the "distribution" with 20 elements, all of which with value 5, will be considered.

Output:

JSON object containing 'status', 'data', 'message'

Create Distribution Profile output JSON

```
{
    "data": [
        { "id": 5,
          "name": "S Curve new"
        }
     ],
    "message": [
        {
            "name": "S Curve new"
            "status": 200,
            "message": "success"
        },
        {
            "name": "Front Loaded"
            "status": 1337,
            "message": "Distribution Profile already exist"
        }
    ],
    "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Get Distribution Profiles

GET /ws/rest/service/v1/cashflow/profile

Purpose:

Get Distribution profiles of company

Input:

All parameters should be URL encoded.

Path Parameter

url parameter(Optional)

filter=

```
{
 "names":["Linear" , "L Curve"]
 }
```

Filter can be used to filter on profile name to get the distribution profile data. If profile does not exist with the given name then that name will be ignored. Profiles of other names will be returned.

Output:

JSON object containing 'status', 'data', 'message'

Get Distribution Profile Output JSON


```
{
    "data": [
        {
            "name": "Linear",
            "status": "Active",
            "distribution": [
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
                "5.0",
```

```
                "5.0",
                "5.0"
            ]
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Status codes are:

1> 200 , for success

Update Distribution Profiles

PUT /ws/rest/service/v1/cashflow/profile

Purpose:

Update the distribution profiles of company.

Input:

Both input & output in JSON format in the body.

Update Distribution Profile input JSON


```
{
    "data": [
        {
            "id": 1,
            "name": "S Curve new",
            "status": "Active",
            "distribution": [
                "0.5",
                "0.5",
                "1.5",
                "1.5",
                "4.0",
                "4.0",
```

```
            "7.5",
            "7.5",
            "11.5",
            "11.5",
            "11.5",
            "11.5",
            "7.5",
            "7.5",
            "4.0",
            "4.0",
            "1.5",
            "1.5",
            "0.5",
            "0.5"
        ]
    },
    {
        "id": 2,
        "name": "Front Loaded",
        "status": "Active",
        "distribution": [
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "6.5",
            "3.5",
            "3.5",
            "3.5",
```

```
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5",
                "3.5"
            ]
        }
    ]
}
```

Supported values for "status" is "Active" or "Inactive".

The "distribution" must be a list of at least 20 elements, the sum of them all must be 100.

Output:

JSON object containing 'status', 'data', 'message'

Update Distribution Profile ouput JSON

```
{
    "data": [],
    "message": [
        {
            "id" : 1,
            "name": "S Curve new"
            "status": 200,
            "message": "success"
        },
        {
            "id" : 2,
            "name": "Front Loaded"
            "status": 200,
            "message": "success"
        }
    ],
    "status": 200
```

```
}
```

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Delete Distribution Profiles

DELETE /ws/rest/service/v1/cashflow/profile

Purpose:

Delete distribution profiles of company

Input:

Both input & output in JSON format in the body

Delete Distribution Profile input JSON

```
{
    "data": {
        "names": ["Linear", "L Curve", "S Curve"]
    }
}
```

Output:

JSON object containing 'status', 'data', 'message'

Delete Distribution Profile Output JSON

```
{
    "data": [],
    "message": [
        {
            "name": "Linear"
            "status": 200,
            "message": "success"
        },
        {
            "name": "L Curve"
            "status": 1337,
            "message": "Distribution Profile do not exist"
        },
```

```
        {
            "name": "S Curve"
            "status": 1338,
            "message": "Distribution Profile is used in Cashflow Curves"
        }
    ],
    "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3000, for Partial success.

Refresh Cashflow Curves

POST /ws/rest/service/v1/cashflow/prop/refresh/{project_number}

Purpose:

Refresh selected CashFlow properties.

The input JSON shall provide various options to be considered for refreshing Curves.

Input:

All parameters should be URL encoded.

project_number: (Required) Specify the Project number in which the curve exists.

POST body is a JSON

Note: POST call has input & output both as JSON in the body.

```
{
"options":{
"property_name": "CBS CF"
}
}
```

Supported options properties

"property_name": "CBS CF"

"rollup_status"<<Active><Inactive>>

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_levelvalue provided is other than above mentioned values, invalid options message will be thrown.

"include_curves":<< Yes/ No>> <!-- default is Yes if not provided→>

If include_curves value provided is other than above mentioned values, invalid options message will be thrown.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Refresh CashFlow Sample


```
{
    "data": [
        {
            "id": 13,
            "name": "Project CF",
            "_record_status": "SUCCESS"
        },
        {
            "id": 14,
            "name": "Project CF FP",
            "_record_status": "SUCCESS"
        },
        {
            "id": 15,
            "name": "xyz",
            "_record_status": "SUCCESS"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```

Get Cashflow Permissions

GET /ws/rest/service/v1/cashflow/prop/permission/{project_number}

Purpose:

Fetch Cash flow Permissions

The input JSON shall provide various options to be considered for fetching cashflow permissions.

Input:

All parameters should be URL encoded.

project_number: (Required) Specify the Project number in which the curve exists.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Refresh CashFlow


```
"data":
 [
   {
     "property_name":"Project CF",
     "user_group":
     [
       {
         "name":"Admin",
         "type":"U",
         "id":1000,
         "permission":
         {
           "modify_permission":1,
           "edit_data":1,
           "view":1
         }
       },
       {
         "name":"test Group",
```

```
          "type":"PU",
          "id":1000014,
          "permission":
          {
            "modify_permission":0,
            "edit_data":1,
            "view":1
          }
        }
      ]
    },
    {
      "property_name":"xyz",
      "user_group":
      [
        {
          "name":"Admin",
          "type":"U",
          "id":1000,
          "permission":
          {
            "modify_permission":1,
            "edit_data":1,
            "view":1
          }
        }
      ]
    }
  ],
  "message":
  [ "success"
  ],
  "status":200
}
```

REST API Details in Schedule Sheet

- Authorization
- Get Schedule Sheet Data Mapping
- Get Schedule Sheets
- Update Schedule Activities
- Response Codes

In This Section

Get Schedule Sheet Data Mapping.....445
Get Schedule Sheets.....446
Update Schedule Activities.....447
Response Codes.....448

Get Schedule Sheet Data Mapping

GET /ws/rest/service/v1/schedule/mapping/{project_number}

Purpose:

Get all list of data mappings defined for this schedule sheet.

Input:

Path Parameter

project_number(Required): Specify the project/shell number.

Url Parameter

sheetname(Required): name of the schedule sheet.

filter (Optional): Filter condition to be used when retrieving the data mapping. It is a JSON object with the below information:

"name"(Optional): Name of data mapping

"isdefault"(Optional): Specify "yes" to get only the default data mapping and "no" to get all the data mappings. Default is "no"

Output:

JSON object containing 'status', 'data', 'message'

Amessage will be present if the status is not 200 otherwise it will be "success".

Sample Response

```
{
   "data":[
      {
        "name":"p6_mapping",
```

```
        "isdefault":"true"
     },
    {
        "name":"stage_mapping",
        "isdefault":"false"
     }
  ]
}
```

Get Schedule Sheets

GET /ws/rest/service/v1/schedule/{project_number}

Purpose:

Get the list of schedule sheet attributes defined in the shell/project.

Input:

Path Parameter

project_number(Required): Specify the project/shell number.

Url Parameter

filter (Optional): Filter condition to be used when retrieving the schedule sheet. It is a JSON object with the below information:

"status": Status of the schedule sheet that needs to be retrieved. Possible values are "active" and "inactive".

"main_sheet": Specify "true" to get the main schedule sheet or "false" to get other non-main schedule sheets.

"sheet_lock": Specify "true" to get the schedule sheets that are locked, "false" to get non locked schedule sheet.

"name": Specify the name of the schedule sheet that needs to be returned in the response.

> **Note**: All fields values of filter condition are case insensitive except the value of "name" field

Sample input parameter

```
{
   "filter":
      {
        "name":"Gas pipeline schedule",
        "status": "open"
```

```
    }
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Sample Response

```
{
  "data": [],
  "message": [
    "Invalid Filter Condition : status"
  ],
  "status": 811
}
```

Update Schedule Activities

PUT /ws/rest/service/v1/schedule/file/{project_number}

Purpose:

Update schedule sheet activities using P6 xml file as an attachment to this service.

Input:

Path Parameter

project_number(Required): Specify the project/shell number.

POST body is a JSON with the below options

Sample Request

```
{
  "data": {
    "sheetName": "Gas pipeline schedule",
    "iszipfile": "yes"
  },
  "scheduleOptions": {
    "MapName": "p6_mapping",
    "ActivityIdentifier": "Id",
    "WBS_NoOfLevelsToMerge": "5",
    "WBS_StartMergeFrom": "2",
```

```
    "ActivityDeletion": "Auto",
    "ActivityFileName": "activity.xml",
    "SpreadFileName": "spreaddfile.xml",
    "CBS_FullPath":"Yes",
    "is_cumulative":"no"
  },
  "_attachment": {
    "file_name": "pipeline_schedule_p6.zip",
    "file_size": "746089",
    "file_content": "base64_encoded string of create.zip file"
  }
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Sample Response

```
{
    "data": [],
    "message": [
        "success"
    ],
    "status": 200
}
```

Response Codes

Code	Message	Details
1400	Schedule Sheet operations are not possible on Generic type of Project	Not available.

REST API Details in Exchange Rates

In This Section

Get Exchange Rates 449
Update Exchange Rates..... 450

Get Exchange Rates

GET /ws/rest/service/v1/exchange

Purpose:

Get Company Exchange Rates

Output:

JSON object containing 'status', 'data', 'message'

Get Exchange Rates ouptut JSON


```
{
    "data":{
    "base_currency": "Indian Rupee (INR)",
    "exchange_rates": [
          {
            "currencyid":2,
            "rate":101.0,
            "currency_name":"Afghani (AFN)",
            "currency_code":"AFN"
          },
          {
            "currencyid":12,
            "rate":211.0,
            "currency_name":"Barbados Dollar (BBD)",
            "currency_code":"BBD"
          },
          {
            "currencyid":40,
            "rate":134.0,
```

```
            "currency_name":"Algeria Dinar (DZD)",
            "currency_code":"DZD"
          }
          ]
    },
    "message": [
      "success"
    ],
    "status": 200
}
```

Status codes are:

1 > 200, for success

Update Exchange Rates

PUT /ws/rest/service/v1/exchange

Purpose:

To add or modify currency rates of active exchange rates set.

Input:

Both input & output in JSON format in the body.

'currency_code' is required in the input request.

Update Exchange Rates input JSON

```
{
  "data":[
  {
    "currencyid":2,
    "rate":110.0,
    "currency_name":"Afghani (AFN)",
    "currency_code":"AFN"
  },
  {
    "currencyid":3,
    "rate":12.0,
```

```
    "currency_name":"Lek (ALL)",
    "currency_code":"ALL"
  }
  ]
}
```


Output:

JSON object containing 'status', 'data', 'message'

Update Exchange Rates output JSON


```
{
  "data":[
  {
    "currencyid":2,
    "rate":110.0,
    "currency_name":"Afghani (AFN)",
    "currency_code":"AFN"
  },
  {
    "currencyid":3,
    "rate":12.0,
    "currency_name":"Lek (ALL)",
    "currency_code":"ALL"
  }
  {
    "currencyid":40,
    "rate":134.0,
    "currency_name":"Algeria Dinar (DZD)",
    "currency_code":"DZD"
  },
  {
     "currencyid":12,
     "rate":211.0,
     "currency_name":"Barbados Dollar (BBD)",
     "currency_code":"BBD"
```

```
  },
  ],
  "message": [
    "success"
  ],
  "status": 200
}
```

Notes:

- Whenever request is made to this service, a new exchange rate will be created (with currencies of currently active exchange rate set), and this new exchange rate will be activated immediately.
- If currency object given in input request does not exist in the active exchange rate set, then this currency object will be added to existing currency list.
- If currency object given in input request already exists in currently active exchange rate set, then the currency rate will be modified with the value given in input object.

Status codes are:

1 > 200, for success

REST API Details in Data Structure Setup

In This Section

Authorization 453
Get Data Elements 453
Create Data Element 455
Update Data Element 457
Delete Data Elements 460
Create DDS Definition 461
Update DDS Definition 464
Delete DDS Definition 467
Get DDS Definition 468
Create DDS Data 470
Update DDS Data 473
Delete DDS Data 475
Get DDS Data 476
Get Data Definition 478
Create Data Definition 483
Update Data Definition 493
Delete Data Definition 500
Response Error Codes 501

Authorization

Refer to the **Authentication** section in the ***REST Services V1*** (on page 171) chapter, above.

Ensure that the integration user has the required Company Administration Permissions.

Get Data Elements

GET /ws/rest/service/v1/ds/data-elements

Version: 20.10

Purpose:

To get list of custom data elements

Input:

All parameters should be URL encoded.

Path Parameter

```
filter =
{
    "data_element": "",
```

```
    "data_definition": "Decimal Amount" ,
    "form_label": "" ,
    "description": "" ,
    "tooltip": ""
}
```

Notes:

- All field values in the filter object are case insensitive
- If filter object is not provided in the request then all data elements list will be returned
- Condition type contains will be used to fetch the data elements
- Filter condition will be applied only if the provided filter values are not-null/non-empty.

Output:

JSON object containing 'status', 'data', 'message'

Get Data Element ouptut JSON

```
{
  "data": [
    {
      "data_element": "sampleDE",
      "data_definition": "Decimal Amount",
      "form_label": "Sample Decimal Element",
      "description": "Test DE",
      "tooltip": null,
      "decimal_format": "5",
      "Pre-Defined Category": null
     },
    {
      "data_element": "sampleDE2",
      "data_definition": "Decimal Amount",
      "form_label": "Sample Decimal Element 2",
      "description": "Decimal DE",
      "tooltip": null,
      "decimal_format": "6",
      "Pre-Defined Category": null
     }, ...
  ],
```

```
  "message": [
      "success"
  ],
  "status": 200
}
```

Status codes are:

1> 200, for success

Create Data Element

POST /ws/rest/service/v1/ds/data-elements

Version: 20.10

Purpose:

To create custom data elements

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create Data Element input JSON

```
{
  "data": [
    {
      "data_element": "sampleDE",
      "data_definition": "Decimal Amount",
      "form_label": "Sample Decimal Element",
      "description": "Test DE",
      "decimal_format": "5"
    },
    {
      "data_element": "sampleImgDE",
      "data_definition": "Image Picker",
      "form_label": "Sample Image Picker",
      "description": "Test Image DE",
      "tooltip": "Test Image DE"
    }
```

```
  ]
}
```

Input JSON field description:

- **Required Fields:**
  - data_element (Required): specify the unique date element name
  - data_definition (Required): specify the field definition to create data element of that type
  - form_label (Required): specify DE Label name
- **Non-required Fields:**
  - Description
  - Tooltip
- **Valid Fields based on the data_definition value:**

data_definition	Required Field Name
Image Picker	height (required)
SYS Rich Text	height (required)
textarea (as data_source)	no_of_lines (required)
Decimal Amount	decimal_format (not required - default value is '8')
SYS Numeric Query Based	hide_currency_symbol (not required - valid values are Yes/No - default value will be No)

Output:

JSON object containing 'status', 'data', 'message'

Create Data Element ouptut JSON

```
{
  "data": [
    {
      "data_element": "sampleDE",
      "data_definition": "Decimal Amount",
      "form_label": "Sample Decimal Element",
      "description": "Test DE",
      "tooltip": null,
      "decimal_format": "5" ,
      "Pre-Defined Category": null
     }
  ],
  "message": [
```

```
    {
      "data_element": "sampleDE",
      "message": "success"
    },
    {
      "data_element": "sampleImgDE",
      "message": "height is required",
      "status": 505
    }
  ],
  "status": 3000
}
```


Status codes are:

1> 200, for success

2> 3000, for partial create.

3> 3002, for invalid JSON input

4> 1101, for Empty or Invalid JSON data

Update Data Element

PUT /ws/rest/service/v1/ds/data-elements

Version: 20.10

Purpose:

To update custom data elements

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update Data Element input JSON


```
{
  "data": [
    {
      "data_element": "sampleDE",
      "data_definition": "SYS Rich Text",
      "form_label": "Sample Rich Text Element",
```

```
      "description": "Test DE",
      "height": "50"
    },
    {
      "data_element": "sampleImgDE",
      "data_definition": "Image Picker",
      "form_label": "Sample Image Picker",
      "description": "Test Image DE",
      "tooltip": "Test Image DE"
    }
  ]
}
```

Input JSON field description:

- **Required Fields:**
  Data_element(Required): specify the existing date element name
- **Non-required Fields:**
  - Description
  - Tooltip
- **Valid Fields based on the data_definition value:**

data_definition	Required Field Name (Existing values will be used if there are no values in the request body)
Image Picker	height (required)
SYS Rich Text	height (required)
textarea (as data_source)	no_of_lines (required)
Decimal Amount	decimal_format (not required - default value is '8')
SYS Numeric Query Based	hide_currency_symbol (not required - valid values are Yes/No - default value will be No)

Notes:

- data_element field value will be used to uniquely identify the existing Data Element.
- Field values can be cleared by providing the empty string in the input request.
- data_definition and form_lable values cannot be emptied/cleared and if no values are provided in the request then already existing values will be taken.
- data_definition field value of a DE cannot be updated if that DE is pre-defined or already deployed.
- For all the fields mentioned above, if no values are provided in the input request then existing values of the given data element will be taken.

Output:

JSON object containing 'status', 'data', 'message'

Update Data Element ouptut JSON


```
{
  "data": [
    {
      "data_element": "sampleDE",
      "data_definition": "SYS Rich Text",
      "form_label": "Sample Rich Text Element",
      "description": "Test DE",
      "tooltip": null,
      "height": "50" ,
      "Pre-Defined Category": null
     }
  ],
  "message": [
    {
      "data_element": "sampleDE",
      "message": "success"
    },
    {
      "data_element": "sampleImgDE",
      "message": "sampleImgDE does not exists",
      "status": 505
    }
  ],
  "status": 3000
}
```


Status codes are:

1> 200, for success

2> 3000, for partial create.

3> 3002, for invalid JSON input

4> 1101, for Empty or Invalid JSON data

Delete Data Elements

DELETE /ws/rest/service/v1/ds/data-elements

Version: 20.10

Purpose:

Deletes custom data elements

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Delete Data Element

```
{
"data":{
    "data_elements": ["simpleDE1", "simpleDE2", "simpleDE3",
"uuu_P6CSIndex"]
    }
}
```

Output:

JSON object containing 'status', 'data', 'message'

Delete Data Element

```
{
    "data": [],
    "message": [
        { "data_element": "SimpleDE1","status":
"200","message":"success"}  ,
        { "data_element": "SimpleDE2","status": "1521","message":"Data
Element is used in Unifier, it cannot be deleted."} ,
        { "data_element": "SimpleDE3","status": "1522","message":"This
data element is being used in uDesigner and cannot be deleted."} ,
        { "data_element": "uuu_P6CSIndex","status":
"1523","message":"Pre-Defined Data Elements cannot be deleted."}
    ],
    "status": 3000
}
```

Status codes are:

1> 200, for success

2> 3000, for partial delete

Create DDS Definition

POST /ws/rest/service/v1/ds/dds

Version: 20.10

Purpose:

To create Dynamic Data Set Definition

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create Dynamic Data Set input JSON

```
{
 "data": [
          {
            "dds_name": "sampleDDS",
            "master_de_name": "countryDE",
            "dds_desc": "sampleDDS",
            "value_set":[
               {
                "data_element":"stateDE",
                 "order":1
                 },
                {
                 "data_element":"cityDE",
                 "order":2
                }
               ],
              "behavior_set":[
                {
                "data_element":"zipcodeDE"
                }
               ]
            },
            {
```

```
            "dds_name": "sampleDDS2",
            "master_de_name": "countryDE",
            "dds_desc": "sampleDDS",
            "value_set":[
               {
                "data_element":"stateDE",
                "order":2
                },
                {
                 "data_element":"cityDE",
                 "order":1
                 }
               ],
              "behavior_set":[
                {
                "data_element":"zipcodeDE"
                }
               ]
            }
        ]
}
```

Input JSON field description:

- **Required Fields:**
  - dds_name: specify the unique DDS name
  - master_de_name: specify Master DE
  - data_element in value_set is required if the value_set is part of the input request
  - order: order of the value set DEs
- **Non Required Fields:**
  - description

Output:

JSON object containing 'status', 'data', 'message'

Create Dynamic Data Set ouptut JSON

```
{
   "data": [
```

```
            {
             "id":12,
             "dds_name": "sampleDDS2",
             "master_de_name": "countryDE",
             "dds_desc": "sampleDDS",
             "value_set":[
               {
               "data_element":"stateDE",
                "order":1
                },
               {
                "data_element":"cityDE",
                "order":2
               }
               ],
               "behavior_set":[
                {
                 "data_element":"zipcodeDE"
                }
                ]
               }]
},
  "message": [
                  {
                      "dds_name": sampleDDS
                      "message": Name already exists. Enter a different
name.,
                      "status": 1500
                  },
                  {
                      "dds_name": "sampletest1",
                      "message": "usrServiceCategoryPD122",
                      "status": 200
                  }
```

```
  ],
  "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Update DDS Definition

PUT /ws/rest/service/v1/ds/dds

Version: 20.10

Purpose:

To Update Dynamic Data Set Definition. action Json value 'delete' will used to remove any value or Behavior set data element for DDS Definition .

Previously existing value set or Behavior set data elements will be reordered based on latest order provided in values set/Behavior set in update DDS definition.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update Dynamic Data Set input JSON

```
{
 "data": [
          { "id":12,
            "dds_name": "sampleDDS",
            "master_de_name": "countryDE",
            "dds_desc": "desc",
            "value_set":[
               {
                "data_element":"stateDE",
                "order":1
                },
                {
                 "data_element":"cityDE",
                 "action":"delete"
```

```
                },
                {
                 "data_element":"address",
                 "order":2
                }
               ],
              "behavior_set":[
                {
                "data_element":"zipcodeDE"
                }
               ]
            },
            {
            "dds_name": "sampleDDS2",
            "master_de_name": "countryDE",
            "dds_desc": "desc",
            "value_set":[
               {
                "data_element":"stateDE",
                "order":1
                },
                {
                 "data_element":"cityDE",
                 "order":2
                }
               ],
              "behavior_set":[
                {
                "data_element":"zipcodeDE"
                }
               ]
            }
        ]
}
```

Input JSON field description:

- **Required Fields:**
  id : specify the DDS id which needs to be updated
  Note: action: "delete" required to delete value set or behavior set data element in DDS Definition

Output:

JSON object containing 'status', 'data', 'message'

Update Dynamic Data Set ouptut JSON


```
{
   "data": [
           {
            "id":12,
            "dds_name": "sampleDDS",
            "master_de_name": "countryDE",
            "dds_desc": "desc",
            "value_set":[
              {
              "data_element":"stateDE",
               "order":1
               },
              {
               "data_element":"address",
               "order":2
              }
              ],
              "behavior_set":[
               {
                "data_element":"zipcodeDE"
               }
               ]
              }]
        },
   "message": [
               {
```

```
                    "dds_name": "sampleDDS"
                    "message": "usrServiceCategoryPD122",
                    "status": 200
                },
                {
                    "dds_name": "sampleDDS2",
                    "message": Name already exists. Enter a different
name.,
                    "status": 1500

                }
  ],
  "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Delete DDS Definition

DELETE /ws/rest/service/v1/ds/dds

Version: 20.10

Purpose:

To delete Dynamic Data Set Definition

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Delete Dynamic Data Set input JSON

```
{
 "data": {
   "dds_name": ["sampleDDS","sampleDDS2"]
  }
}
```

Input JSON field description:

- **Required Fields:**
  dds_name: DDS name

Output:

JSON object containing 'status', 'data', 'message'

Delete Dynamic Data Set ouptut JSON

```
{
    "data": {

    },
    "message": [

          { name: "SampleDDS","status": "200","message":"success"},
          { name: "SampleDDS2","status": "1509","message":"DDS does not
exist"}

    ],
    "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Get DDS Definition

GET /ws/rest/service/v1/ds/dds

Version: 20.10

Purpose:

To get Dynamic Data set Definition

Input:

All parameters should be URL encoded.

URL query parameter -

```
filter=
{
"dds_name":"Sample DDS"
```

```
}
```

Supported options in filter:

- dds_name: specify the DDS name
- master_de_name: specify Master DE
- dds_desc: Description of the DDS
- master_de_label: Form Label of the Master DE

Note: If no filter options provided service will return all DDS .

Both input & output in JSON format in the body

Get Dynamic Data Set input filter

```
"filter":
         {
           "dds_name": "sampleDDS",
           "master_de_name": "countryDE",
           "dds_desc": "sampleDDS",
           "master_de_label": "my sample"
           }
```

Input JSON field description:

- Non Required Fields:
  - dds_name: specify the DDS name
  - master_de_name: specify Master DE
  - dds_desc: Description of the DDS
  - master_de_label: Form Label of the Master DE

  It is a "contains" and case insensitive search.

Output:

JSON object containing 'status', 'data', 'message'

Get Dynamic Data Set ouptut JSON

```
{
   "data": [
           {
            "id":12,
            "dds_name": "sampleDDS",
            "master_de_name": "countryDE",
            "dds_desc": "sampleDDS",
            "master_de_label": "my sample"
            "value_set":[
```

```
                {
                "data_element":"stateDE",
                 "order":1
                 },
                {
                 "data_element":"cityDE",
                 "order":2
                }
                ],
                "behavior_set":[
                 {
                  "data_element":"zipcodeDE"
                 }
                 ]
                }]
},
  "message":  [
     "Success"
  ],
  "status":200
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Create DDS Data

POST /ws/rest/service/v1/ds/ddsdata

Version: 20.10

Purpose:

To create data set for a given Dynamic Data Set

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create DDS Data input JSON

```
{   "data": {
        "dds_name": "mydds",
        "data_set": [{
                "masterDE":"value2",
                "value_set": {
                    "valueDE1": "value1",
                    "valueDE2": "value2"
                },
                "behavior_set": {
                    "behavede1": "disabled"
                }
            },
            {    "masterDE":"value1",
                "value_set": {
                    "valueDE1": "value1",
                    "valueDE2": "value3"
                },
                "behavior_set": {
                    "behavede1": "Required"
                }
            }
        ]
    }
}
```

Input JSON field description:

- **Required Fields:**
  - dds_name(Required): specify the dynamic data set for which the data needs to be created
  - value set(Required if configured): specify the values for the DEs defined as value
- **Non Required Fields:**
  - behavior set(required if configured): specify the value for the DEs defined as behavior

Output:

JSON object containing 'status', 'data', 'message'

Create DDS data output JSON

```
{
    "data": {
        "dds_name": "mydds",
        "data_set": [{
                "id": 710,
                "masterDE":"value1",
                "value_set": {
                    "dename1": "value",
                    "dename2": "value"
                },
                "behavior_set": {
                    "dename3": "disabled"
                }
            },
            {
                "id": 711,
                "masterDE":"value2",
                "value_set": {
                    "dename1": "value",
                    "dename2": "value"
                },
                "behavior_set": {
                    "dename3": "disabled"
                }
            }
        ]
    },
    "message":  "success"   ,
    "status": 200
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Update DDS Data

POST /ws/rest/service/v1/ds/ddsdata

Version: 20.10

Purpose:

To update data set for a given Dynamic Data Set

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update DDS Data input JSON

```
{   "data": {
        "dds_name": "mydds",
        "data_set": [{
                "id": 710,
                "masterDE":"value1updated",
                "value_set": {
                    "valueDE1": "value1updated",
                    "valueDE2": "value2updated"
                },
                "behavior_set": {
                    "behavede1": "disabled"
                }
            },
            {
                "id": 711,
                "countryDE":"value2updated",
                "value_set": {
                    "valueDE1": "value1",
                    "valueDE2": "value3"
                },
```

```
                "behavior_set": {
                    "behavede1": "Required"
                }
            }
        ]
    }
}
```

Input JSON field description:

- **Required Fields:**
  - dds_name: specify the dds name for which the data needs to be updated
  - id: specify the dda data row id to uniquely identify the data set

Output:

JSON object containing 'status', 'data', 'message'

Update DDS data output JSON

```
{
    "data": {
        "dds_name": "mydds",
        "data_set": [{
                "id": 710,
                "masterDE":"value1updated",
                "value_set": {
                    "dename1": "value",
                    "dename2": "value"
                },
                "behavior_set": {
                    "dename3": "disabled"
                }
            },
            {
                "id": 711,
                "masterDE":"value1updated",
                "value_set": {
                    "dename1": "value",
                    "dename2": "value"
```

```
                },
                "behavior_set": {
                    "dename3": "disabled"
                }
            }
        ]
    },
    "message":  "success"   ,
    "status": 200
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Delete DDS Data

DELETE /ws/rest/service/v1/ds/ddsdata

Version: 20.10

Purpose:

To delete DDS data

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Delete DDS data input JSON

```
{
 "data": {
   "dds_name": "sampleDDS",
   "id":[712,714]
  }
}
```

Input JSON field description:

- **Required Fields:**
  - dds_name: specify the DDS name of the ddds data set to be deleted. Multiple DDS can be deleted in one request.

- id: id of the DDS data set.
  Note: Use id vs value set and behavior set combination

Output:

JSON object containing 'status', 'data', 'message'

Delete DDS data output JSON

```
{
    "data": {

    },
    "message": [

          { id: "712","status": "200","message":"success"},
          { id: "714","status": "1509","message":"DDS does not exist"}

    ],
    "status": 3000
}
```

Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Get DDS Data

GET /ws/rest/service/v1/ds/ddsdata

Version: 20.10

Purpose:

To get DDS data

Input:

All parameters should be URL encoded.

URL query parameter -

dds_name="sample dds"

filter=

{

"masterDE":"val1",

"value_set":{ "leed":"test","uuu_state":"TS"}

"behavior_set":{"uuu_name":"Required","uuu_desc":"Disabled"}

}

Supported options in filter :

master_de_value: specify Master DE value

value_de: specify the value de with the values if configured

behavior_de :specify the value of behavior de with the values if configured

Both input & output in JSON format in the body

Get DDS data input filter

"dds_name": "sampleDDS",

"filter":

{

"countryDE":"IN",

"value_set":{ "leed":"test","uuu_state":"TS"}

}

Input JSON field description:

- **Required Fields:**
  - dds_name:specify the name of the dds
- **Non Required Fields:**
  - master_de_value: specify Master DE value
  - value_de1_value: Value of the value_de if configured
  - behavior_de1_value: Value of the behavior_de if configured

It is a equals and case insensitive search

Note: If more than one Value DE is configured, it can be used in filter using value_de2

Output:

JSON object containing 'status', 'data', 'message'

Get Dynamic Data Set output JSON

```
{
    "data": [{
        "id":720,
        "dds_name": "sampleDDS",
        "data_set": [{
                "id": 710,
```

```
                "CountryDE":"IN",
                "value_set": {
                    "leed": "test",
                    "uuu_state": "TS"
                },
                "behavior_set": {
                    "contract_name": "disabled"
                }
                }]
          }],
  "message":  [
     "Success"
  ],
  "status":200
}
```


Status codes are:

1> 200 , for success

2> 3002, for invalid JSON input

3> 1101, for Empty or Invalid JSON data

Get Data Definition

GET /ws/rest/service/v1/ds/data-def

Version: 20.11

Purpose:

To get list of data definitions

Input:

All parameters should be URL encoded.

Path Parameter

type = Basic / Cost Codes / Data Picker

filter =

{

"name": "Sample DD Name",

"data_source": ""

```
        }
```

Notes:

- All field values in the filter object are case insensitive type is case insensitive.
- If filter object is not provided in the request then all data definitions list will be returned
- Condition type contains will be used to fetch the data definitions
- Filter condition will be applied only if the provided filter values are not-null/non-empty.
- type: possible values will be Basic, Cost Codes, Data Picker
- name: Name of the Data definition
- data_source: Data source of the data picker. Only applicable for data picker type.

Output:

JSON object containing 'status', 'data', 'message'

Basic type Output:

Get Data Definition of type Basic output JSON


```
{
 "data": [
        {
            "data_size": 0,
            "name": "Account Code Picker",
            "data_type": "Integer",
            "input_type": "Picker",
            "used": "Yes",
            "category": "System"
        },
        {
            "data_size": 128,
            "name": "Account Type PD",
            "data_type": "String",
            "input_type": "Pull-down Menu",
            "data_set": [],
            "used": "Yes",
            "category": "Company"
        },
        {
            "data_size": 128,
            "name": "Action Required PD",
```

```
            "data_type": "String",
            "input_type": "Pull-down Menu",
            "dataset_non_modifiable": "No",
            "data_set": [
                {
                    "status": "Active",
                    "value": "No",
                    "label": "No",
                    "row_num": 1,
                    "is_default": "No"
                },
                {
                    "status": "Active",
                    "value": "Yes",
                    "label": "Yes",
                    "row_num": 2,
                    "is_default": "No"
                }
            ],
            "used": "Yes",
            "category": "^Company"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Cost Code type Output:

Get Data Definition of type Cost Codes output JSON


```
{
 "data": [{
            "data_size": 32,
```

```
            "name": "Segment 1",
            "data_type": "String",
            "input_type": "Text Box",
            "used": "Yes",
            "label": "Segment 1",
            "category": "CBS Code"
        },
        {
            "data_size": 32,
            "name": "segment 2",
            "data_type": "String",
            "input_type": "Pull-down Menu",
            "label": "Segment 2",
            "dataset_non_modifiable": "Yes",
            "data_set": [
                {
                    "status": "Active",
                    "value": "Yes",
                    "label": "Yes",
                    "row_num": 1,
                    "is_default": "No"
                },
                {
                    "status": "Active",
                    "value": "No",
                    "label": "No",
                    "row_num": 2,
                    "is_default": "No"
                }
            ],
            "used": null,
            "category": "CBS Code"
        }
    ],
```

```
    "message": [
        "success"
    ],
    "status": 200
}
```


DataPicker type output:

Get Data Definition of type Data Pickers output JSON


```
{
 "data": [
        {
            "data_element": "uuu_user_name",
            "display_element": "Name",
            "name": "Action Item Approver DP",
            "used": "Yes",
            "category": "User Attributes",
            "data_source": "User Attributes"
        },
        {
            "data_element": "space_sp_space_name",
            "display_element": "Space Name",
            "name": "Usable Space DP",
            "used": "Yes",
            "category": "Space Manager",
            "data_source": "Usable Space"
        },
        {
            "data_element": "ugenPropertyName",
            "display_element": "Property Name",
            "name": "Property DP",
            "used": "Yes",
            "category": "Business Process",
            "data_source": "Prospective Properties"
        },
```

```
        {
            "data_element": "uveVendorNameTB50",
            "display_element": "Vendor Name",
            "name": "Vendors PK",
            "used": "Yes",
            "category": "Business Process",
            "data_source": "Vendors"
        }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Status codes are:

1> 200, for success

Create Data Definition

POST /ws/rest/service/v1/ds/data-def

Version: 20.11

Purpose:

To create data definition

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Create DD Basic type

Input JSON field description:

Required Fields:

- name: specify the data definition name
- type: Basic
- data_type: Integer/String are possible values.
- input_type: Text Box, Pull-down Menu, Radio Buttons, Multi-select Input, Multiple Text Lines, Checkbox are possible values based on the data type.
- data_size: has to be provided for Text Box and Multiple Text Lines.

Non-required Fields:

- default_value: Default value can be set.
- data_set: Value and Labels for Pull-down Menu , Radio buttons, Multi-select Input
- dataset_non_modifiable: Yes/No.

Valid Fields based on the data_definition:

data_type	input_type	datasize	separator	data_set
Integer	Checkbox	n/a	n/a	n/a
Integer	Pull-down Menu	n/a	n/a	pull down value and labels need to be set
Integer	Radio Buttons	n/a	n/a	value and labels need to be set
String	Text Box	required.	n/a	n/a
String	Multiple Text Lines	required.	n/a	n/a
String	Pull-down Menu	n/a	n/a	pull down value and labels need to be set
String	Radio Buttons	n/a	n/a	value and labels need to be set
String	Multi-select Input	n/a	comma is the separator.	value need to be set

Notes:

Category will be defaulted to "Company" in case of basic type of data definition.

"System" category type of data definition cannot be created.

type is case insensitive.

dataset_non_modifiable is case insensitive.

Basic type input JSON

```
{
 "options":{
                "type": "Basic"
  },
 "data": [{
                "name": "sample basic 1",
                "data_size": "128",
                "data_type": "String",
                "input_type": "Text Box"
```

```
        },
        {
            "name": "sample basic 2",
            "data_type": "Integer",
            "input_type": "Pull-down Menu",
            "dataset_non_modifiable": "Yes",
            "data_set": [{
                    "value": 0,
                    "label": "first",
                    "row_num": 1,
                    "status": "Active",
                    "is_default": "No"
                },
                {
                    "value": 1,
                    "label": "Fixed Amount",
                    "row_num": 2,
                    "status": "Active",
                    "is_default": "Yes"
                },
                {
                    "value": 2,
                    "label": "last",
                    "row_num": 3,
                    "status": "Active",
                    "is_default": "No"
                }
            ]
        }
    ]
}
```


Output:

JSON object containing 'status', 'data', 'message'.

Basic type output JSON


```
{
    "data": [
        {
            "data_size": "128",
            "name": "sample basic 1",
            "data_type": "String",
            "input_type": "Text Box",
            "default_value": "",
            "used": "",
            "category": "Company"
        },
        {
            "data_size": "0",
            "dataset_non_modifiable": "Yes",
            "name": "sample basic 2",
            "data_type": "Integer",
            "input_type": "Pull-down Menu",
            "data_set": [
                {
                    "value": "0",
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "No"
                },
                {
                    "value": "1",
                    "row_num": 2,
                    "status": "Active",
                    "label": "Fixed Amount",
                    "is_default": "Yes"
                },
```

```
                {
                    "value": "2",
                    "row_num": 3,
                    "status": "Active",
                    "label": "last",
                    "is_default": "No"
                }
            ],
            "used": "",
            "category": "Company"
        }
    ],
    "message": [
        {
            "message": "success",
            "data_definition": "sample basic 1",
            "status": 200
        },
        {
            "message": "success",
            "data_definition": "sample basic 2",
            "status": 200
        }
    ],
    "status": 200
}
```


Create DD Cost Code type

Input JSON field description:

Required Fields:

name: specify the field definition name

type: Cost Codes

label : has to be specified.

input_type: Text Box, Pull-down Menu are possible values.

Non-required Fields:

default value: Default value can be set.

data_set: Value and Labels are applicable for Pull down menu.

dataset_non_modifiable: Yes/No.

Notes:

Category will be defaulted to "CBS Code" in case of cost codes type of data definition.

type is case insensitive.

dataset_non_modifiable is case insensitive.

Cost Codes type input JSON


```
{
"options":{
                "type": "Cost Codes"
 },
 "data": [{
                "name": "sample costcode 1",
                "input_type": "Text Box",
                "label": "Test 1"
          },
          {
                "name": "sample costcode 2",
                "input_type": "Pull-down Menu",
                "label": "Test 2",
                "dataset_non_modifiable": "No",
                "data_set": [{
                          "value": "Type 1",
                          "label": "first",
                          "row_num": 1,
                          "status": "Active",
                          "is_default": "Yes"

                     },
                     {
                          "value": "Type 2",
```

```
                    "label": "Fixed Amount",
                    "row_num": 2,
                    "status": "Active",
                    "is_default": "No"

                },
                {
                    "value": "Type 3",
                    "label": "last",
                    "row_num": 3,
                    "status": "Inactive",
                    "is_default": "No"
                }
            ]
        }
    ]
}
```


Output:

JSON object containing 'status', 'data', 'message'

Cost Codes type output JSON


```
{
    "data": [
        {
            "data_size": "32",
            "name": "sample costcode 1",
            "data_type": "String",
            "input_type": "Text Box",
            "default_value": "",
            "used": "",
            "label": "Test 1",
            "category": "CBS Code"
        },
        {
```

```
            "data_size": "32",
            "dataset_non_modifiable": "No",
            "name": "sample costcode 2",
            "data_type": "String",
            "input_type": "^ull-down Menu",
            "data_set": [
                {
                    "value": "Type 1",
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "Yes"
                },
                {
                    "value": "Type 2",
                    "row_num": 2,
                    "status": "Active",
                    "label": "Fixed Amount",
                    "is_default": "No"
                },
                {
                    "value": "Type 3",
                    "row_num": 3,
                    "status": "Inactive",
                    "label": "last",
                    "is_default": "No"
                }
            ],
            "used": "",
            "label": "Test 2",
            "category": "CBS Code"
        }
    ],
```

```
    "message": [
        {
            "message": "success",
            "data_definition": "sample costcode 1",
            "status": 200
        },
        {
            "message": "success",
            "data_definition": "sample costcode 2",
            "status": 200
        }
    ],
    "status": 200
}
```


Create DD Data Picker type

Input JSON field description:

Required Fields:

name: specify the field definition name

category: Business process, Shell Manager, Planning Items, Resource Manager, User Attributes and Space Manager are the system defined categories. You can provide other category values based on the modules loaded in the environment, for ex: Material Inventory Manager.

data_source: name of the data source for the given category

data_element: provide the data element for the selected data source.

type: Data Picker

> **Note**: Type is case insensitive.

Data Picker type input JSON


```
{
 "options":{
            "type": "Data Picker"
 },
 "data": [{
            "name": "sample dp 1",
            "category": "Business Process",
```

```
            "data_source": "Vendors",
            "data_element": "uuu_user_name"
        },
        {
            "name": "sample dp 2",
            "category": "Shell Manager",
            "data_source": "Buildings",
            "data_element": "uBuildingNumber"
        }
    ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Data Picker type output JSON

```
{
    "data": [
        {
            "data_element": "uBuildingNumber",
            "display_element": "Building Code",
            "name": "sample dp 2",
            "used": "",
            "category": "Shell Manager",
            "data_source": "Buildings"
        }
    ],
    "message": [
        {
            "message": "invalid data element provided for Vendors",
            "data_definition": "sample dp 1",
            "status": 619
        },
        {
            "message": "success",
```

```
            "data_definition": "sample dp 2",
            "status": 200
        }
    ],
    "status": 3000
}
```

Status codes are:

1> 200, for success

2> 3000, for partial create.

3> 3002, for invalid JSON input

4> 1101, for Empty or Invalid JSON data

5> 619, for invalid field value

6> 620, for input is required.

Update Data Definition

PUT /ws/rest/service/v1/ds/data-def

Version: 20.11

Purpose:

To update data definition

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

Update Basic Data Defintion:

Input JSON field description:

Required Fields:

- name: specify the data definition name
- type: Basic

Non-required Fields:

- default_value: Default value can be set.
- data_set: Value and Labels for Pull-down Menu , Radio buttons, Multi-select Input
- dataset_non_modifiable: Yes/No.
- dataset_delete_all: Yes

Notes:

- Default value for a data picker can only be updated as part of basic data definition type.

- Sequential ordering of data set is mandatory else it returns an error.
- The data_set which is provided in the input will completely replace the existing data_set. So be sure to get the data_set through GET call and use it for any type of update operations on data_set.
- If data_set is not provided or is empty then existing data_set will not be modified at all.
- Type is case insensitive.
- dataset_non_modifiable is case insensitive.
- dataset_delete_all: if user wishes to remove all data set values then they can use this field. This field is case insensitive.

Update Basic Data Definition input JSON

```
{
  "options":{
            "type": "Basic"
  },
  "data": [
    {
        "name": "sample basic 1",
        "default_value": "Entering default value"
    },
    {
     "name": "sample basic 2",
     "dataset_non_modifiable": "No",
     "data_set": [
                {
                    "value": "0",
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "No"
                },
                {
                    "value": "1",
                    "row_num": 2,
                    "status": "Inactive",
                    "label": "Fixed Amount",
```

```
                    "is_default": "No"
                }
            ]
        }
    ]
}
```


Output:

JSON object containing 'status', 'data', 'message'

Update Basic Data Definition output JSON


```
{
    "data": [
        {
            "data_size": "128",
            "name": "sample basic 1",
            "data_type": "String",
            "input_type": "Text Box",
            "default_value": "Entering default value",
            "used": "",
            "category": "Company"
        },
        {
            "data_size": "0",
            "dataset_non_modifiable": "No",
            "name": "sample basic 2",
            "data_type": "Integer",
            "input_type": "Pull-down Menu",
            "data_set": [
                {
                    "value": "0",
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "No"
```

```
                },
                {
                    "value": "1",
                    "row_num": 2,
                    "status": "Inactive",
                    "label": "Fixed Amount",
                    "is_default": "No"
                }
            ],
            "used": "",
            "category": "Company"
        }
    ],
    "message": [
        {
            "message": "success",
            "data_definition": "sample basic 1",
            "status": 200
        },
        {
            "message": "success",
            "data_definition": "sample basic 2",
            "status": 200
        }
    ],
    "status": 200
}
```


Update Cost Codes Data Defintion:

Input JSON field description:

Required Fields:

- name: specify the data definition name
- type: Cost Codes
- Non-required Fields:
- default_value: Default value can be set.

- data_set: Value and Labels for Pull-down Menu
- dataset_non_modifiable: Yes/No.
- dataset_delete_all: Yes

Notes:

- Sequential ordering of data set is mandatory else it returns an error.
- The data_set which is provided in the input will completely replace the existing data_set. So be sure to get the data_set through GET call and use it for any type of update operations on data_set.
- If data_set is not provided or is empty then existing data_set will not be modified at all.
- Type is case insensitive.
- dataset_non_modifiable is case insensitive
- dataset_delete_all: if user wishes to remove all data set values then they can use this field. This field is case insensitive.

Update Cost Codes Data Definition input JSON

```
{
  "options":{
            "type": "Cost Codes"
  },
  "data": [
    {
        "name": "sample costcode 1",
        "default_value": "Entering default value"
    },
    {
     "name": "sample costcode 2",
     "dataset_non_modifiable": "No",
     "data_set": [
                {
                    "value": "0",
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "No"
                },
                {
                    "value": "1",
```

```
                    "row_num": 2,
                    "status": "Inactive",
                    "label": "Fixed Amount",
                    "is_default": "No"
                }
        ]
    }
  ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Update Cost Codes Data Definition output JSON

```
{
    "data": [
        {
            "data_size": "32",
            "name": "sample costcode 1",
            "data_type": "String",
            "input_type": "Text Box",
            "default_value": "Entering default value",
            "used": "",
            "label": "Test 1",
            "category": "CBS Code"
        },
        {
            "data_size": "32",
            "dataset_non_modifiable": "No",
            "name": "sample costcode 2",
            "data_type": "String",
            "input_type": "Pull-down Menu",
            "data_set": [
                {
                    "value": "0",
```

```
                    "row_num": 1,
                    "status": "Active",
                    "label": "first",
                    "is_default": "No"
                },
                {
                    "value": "1",
                    "row_num": 2,
                    "status": "Inactive",
                    "label": "Fixed Amount",
                    "is_default": "No"
                }
            ],
            "used": "",
            "label": "Test 2",
            "category": "CBS Code"
        }
    ],
    "message": [
        {
            "message": "success",
            "data_definition": "sample costcode 1",
            "status": 200
        },
        {
            "message": "success",
            "data_definition": "sample costcode 2",
            "status": 200
        }
    ],
    "status": 200
}
```


Status codes are:

1> 200, for success

2> 3000, for partial create.

3> 3002, for invalid JSON input

4> 1101, for Empty or Invalid JSON data

5> 619, for invalid field value

6> 620, for input is required.

Delete Data Definition

DELETE /ws/rest/service/v1/ds/data-def

Version: 20.11

Purpose:

Deletes data definition

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body

options: user has to specify the type of data definition, i.e Basic, Cost Codes, Data Picker.

> **Note**: Type is case insensitive.

Delete Data Definition Input JSON

```
{
"options":{
    "type": "Basic"
  },
"data":{
    "data_definitions": ["simple DD1", "simple DD2"]
    }
}
```

Output:

JSON object containing 'status', 'data', 'message'

Delete Data Definition Output JSON

```
{
    "data": [],
    "message": [
```

```
        { data_definition: "Simple DD1","status": "1531","message":"You
can not delete deployed definitions."}  ,
        { data_definition: "Simple DD2","status":
"200","message":"success"}
    ],
    "status": 3000
}
```

Status codes are:

1> 200, for success

2> 3000, for partial delete

3> 1531, for cannot delete data definition.

4> 1532, for data definition does not exists.

5> 3002, for invalid JSON input.

Response Error Codes

The following lists the response error codes for the REST API Details in Data Structure Setup:

Code	Message	Details
200	success	If the request is successful
500	Server Error, contact the system administrator.	If there is server exception when running this service
600	Invalid options	
619	Invalid field value (or) a field-specific error message will be displayed	If an unsupported value is provided for any field
620	Input Required (or) a field-specific error message will be displayed	If the required field value is not provided in the request
1101	Empty or Invalid JSON data	
1500	Required field validation for DDS Name	
1501	Cannot begin with a number.	
1502	Allowed characters are A-Z a-z 0-9 _	
1503	Reserved word validation (Eg: starts with sys,uuu..etc)	
1504	Starts with Underscore validation	
1505	Field Max length validation.	
1506	Value set contains Master data	

	element	
1507	Invalid Order for Value set and Behavior set	
1508	DDS name already Exist	
1509	Invalid Data element in DDS Definition	
1510	DDS usage validation for update and delete DDS Definition.	
1511	Invalid DDS name	
1512	Value Set is Required	
1513	Invalid DDS Data Options	
1514	Option value was doesn't match the available data options	
1515	Duplicate dds data combination.	
1516	Restricted data element that is part of the behavior. The value of this data element cannot be set as Disabled and Clear	
1517	Value Set/Behavior Set contains duplicate DE's for DDS name	
1518	Invalid DDS Data id	
1519	Invalid Action for Update DDS Definition	
1520	data_definition value cannot be updated	If the DE is pre-defined or deployed then data_definition of that DE cannot be updated
1521	Data Element is used in Unifier, it cannot be deleted.	
1522	This data element is being used in uDesigner and cannot be deleted.	
1523	Pre-Defined Data Elements cannot be deleted.	
1531	Cannot delete data definition.	
1532	Data definition does not exists.	
3000	Partial delete	
3002	Invalid Input	

Partner Company Services V1

The following table contains detailed information on Data Elements related to Partner Company REST Services:

Element	Sub-element	UI Label	Data Type	Mandatory?	GET	CREATE	UPDATE	Note
companyname		Name	String	Y	√	√	√	
shortname		Short Name	String	Y	√	√	√	
description		Description	String		√	√	√	
duns		DUNS	String		√	√	√	
url		Home Page URL	String		√	√	√	
helpurl		Help URL	String		√	√	√	
status		Status	Integer		√	√	√	This is optional. If not supplied, inactive company will be created. Possible values are 0 for inactive, and 1 for active.
support_contactemail		Support Contact Info - Email	String		√	√	√	
support_contactphone		Support Contact Info - Phone	String		√	√	√	
support_instructions		Support Contact Info - Instructions	String		√	√	√	
support_emailnotification		Support Contact Info - Email Notifications	String		√	√	√	
elearn_url		eLearning Access - URL	String		√	√	√	
elearn_la		eLearning Access	Stri		√	√	√	

bel		- Label	ng					
elearn_contactemail		eLearning Access - Contact Email	String		√	√	√	
elearn_instructions		eLearning Access - Instructions	String		√	√	√	
addresses					√	√	√	This is optional. If not supplied, one of the owner company address will be added to this company.
	addresstype	At Address Tab - Address Type	Integer	Y	√	√	√	For possible value, refer Unifier integration guide. If not sure, provide 1 for address type "Headquarter".
	addressline1	At Address Tab - Address 1	String	Y	√	√	√	
	addressline2	At Address Tab - Address 2	String		√	√	√	
	addressline3	At Address Tab - Address 3	String		√	√	√	
	city	At Address Tab - City	String	Y	√	√	√	
	state	At Address Tab - State	String	Y	√	√	√	
	zip	At Address Tab - Zip/Postal Code	String	Y	√	√	√	
	country	At Address Tab - Country/Region	String	Y	√	√	√	For list of country codes, refer Unifier integration guide
	phone	At Address Tab - Phone	String		√	√	√	
	fax	At Address Tab - Fax	String		√	√	√	

password policy					√	√	√	This is optional. If not supplied, owner company password policy will be added to this company.
	min_length	At Security Tab - Minimum overall character(s)	Integer	Y	√	√	√	
	max_length	At Security Tab - Maximum overall character(s)	Integer		√	√	√	
	min_numericchar	At Security Tab - Minimum numeric character(s)	Integer		√	√	√	
	min_alpchar	At Security Tab - Minimum alphabetic character(s)	Integer		√	√	√	
	min_specchar	At Security Tab - Minimum special character(s)	Integer		√	√	√	
	chk_uname	At Security Tab - Password cannot be same as Username	Integer		√	√	√	
	chk_flname	At Security Tab - Password cannot be same as First or Last Name	Integer		√	√	√	
	chk_prevpwd	At Security Tab - Password cannot be same as last	Integer		√	√	√	
	pwd_validfor	At Security Tab - Password expiration	Integer		√	√	√	
	inform_exp	At Security Tab - Inform user before expiration	Integer		√	√	√	
	max_attempts	At Security Tab - Maximum login attempts	Integer		√	√	√	
	max_i	At Security Tab -	Inte		√	√	√	

	nactiv e	Suspend inactive user after	ger					

Get Partner Company

Method: GET

Objective: Provide the ability to get partner company information.

Usage:

1) This will be the GET request.
2) The service URL to be used will be `http://<environment url>ws/rest/service/v1/admin/company?shortname=<urlencoded_shortname>`.
3) If the shortname parameter is not specified, then the URL will retrieve the entire list of partner companies.

Output:

If the service runs successfully, the output will be Status, Message, and Data.

Sample output JSON when shortname parameter is passed


```
{
  "data":
  [
    {
      "support_contactphone":null,
      "elearn_url":null,
      "addresses":
      [
        {
          "zip":"94588",
          "country":"US",
          "city":"Pleasanton",
          "phone":null,
          "addressline3":null,
          "addresstype": 1,
          "addressline2":null,
          "addressline1":"5815 Owens Drive",
          "state":"CA",
          "fax":null
        }
      ],
```

```
        "support_emailnotification":null,
        "passwordpolicy":
        {
          "inform_exp":7,
          "min_specchar":1,
          "max_inactive":360,
          "min_length":6,
          "min_numericchar":1,
          "max_attempts":5,
          "min_alpchar":1,
          "pwd_validfor":180,
          "chk_uname":1,
          "chk_prevpwd":0,
          "max_length":0,
          "chk_flname":1
        },
        "helpurl":null,
        "description":null,
        "support_contactemail":null,
        "shortname":"partner01",
        "url":null,
        "companyname":"Partner Company - 01",
        "elearn_instructions":null,
        "elearn_contactemail":null,
        "support_instructions":null,
        "duns":null,
        "elearn_label":null,
        "status":1
      }
    ],
    "message":
    [
    ],
    "status":200
  }
```


Sample Output for all partner companies


```
  {
    "data":
    [
      {
```

```
        "companyname":"Partner Company - 01",
        "shortname":"partner01",
        "status":1
      },
      {
        "companyname":"Partner Company - 02",
        "shortname":"partner02",
        "status":1
      }
    ],
    "message":
    [
    ],
    "status":200
  }
```

Create Partner Company

Method: POST

Objective: Create partner company based on information provided.

Usage:

This will be the POST request.

The service URL to be used will be `http://<environment url>/ws/rest/service/v1/admin/company`

Input:

The user will be required to enter the company information in JSON format.

Minimum data requirements are the mandatory fields when creating partner company or you can specify detailed data, similar to output of GET request.

Sample JSON to create Partner company

```
  {
   "companyname":"Partner Company - 01",
   "shortname":"partner01",
   "addresses": [{
          "addresstype": 1,
          "addressline1":"San Jose CA",
          "city":"San Jose",
          "state":"CA",
```

```
        "zip":"95110",
        "country":“US”
    }],
    "passwordpolicy":{
        "min_length": 3,
        "max_length": 7,
    }
    }
```

Output:

The output will consist of Status, message.

```
   Sample Output
    Sample output JSON
    {
    "data":
    [
    ],
    "message":
    [
        "The partner company has been created"'
    ],
    "status":200
    }
    Error Handling
    {
    "data": [
    "companyname is mandatory to create partner company",
    "passwordpolicy is mandatory to create partner company",
    "addresses is mandatory to create partner company"
    ],
    "message": [
    "One or more records contain errors."
    ],
    "status": 5555
```

```
}
```

The mandatory existing validations that exist for creating partner company will be applicable in REST service as well.

Validations for creating partner company

Status	Condition	Message
200	Success	The partner company has been created.
900	Mandatory field not specified	<data_element> is mandatory to create partner company
500	Server Error	Server Error, contact the system administrator.
901	Invalid data value	Invalid data for <data_element>.
5555	One or more records contain errors.	One or more records contain errors.

Update Partner Company

Method: PUT

Objective: Provide the ability to update partner company based on information provided.

Usage:

This will be the PUT request.

The service URL to be used will be: `http://<environment url>/ws/rest/service/v1/admin/company`

Input:

Company information in JSON format.

Sample JSON to update Partner company

```
{
"companyname":"Partner Company - 01",
"shortname":"partner01",
"addresses": [{
"addresstype": 1,
"addressline1":"488 Almaden Blvd",
"city":"San Jose",
"state":"CA",
"zip":"95110",
"country":"US"
}],
"passwordpolicy":{
```

```
"min_length": 1,
"max_length": 3
}
}
```

Output:

If the service runs successfully, then the return data will be Status and message.

The mandatory existing validations that exist for updating partner company will be applicable in REST service as well

The shortname is the identifier for updating the partner company details.

The other data elements which are not included as part of the update request will retain its value on execution.

Validations for updating partner company

Status	Condition	Message
200	Success	The partner company {partner company name} has been created.
901	Invalid data value	Invalid data for <data_element>.
902	Partner company specified does not exist as shortname specified is not matching.	The partner company does not exist.
5555	One or more records contain errors.	One or more records contain errors.

Update Partner Company Shortname

Method: POST

The Company "shortname" is the identifier.

URL: <unifier_url>/ws/rest/service/v1/admin/company/shortname

Sample Input JSON:

```
{
    "shortname":"partner07",
    "new_shortname":"partner07123"
}
```

Add (Update) Partner Company Shell Membership

Method: PUT

URL: <unifier_url>/ws/rest/service/v1/admin/company/shell/member/add

Objective: Add partner company to a shell based on information provided.

Usage:

This will be a PUT request.

The service URL to be used will be: `http://<environment url>/ws/rest/service/v1/admin/company/shell/member/add`

Input:

Company information in JSON format.

Sample JSON to add member company to a shell

```
{
    "shortname":"partner01",
    "shellnumber":"7896"
}
```

Output:

Return data will be JSON format too.

Validations:

Status	Condition	Message
200	Success	The member company {name} has been added to shell {name}
910	Adding inactive partner company to a given shell.	The inactive partner company {name} cannot be added to shell {name}.
911	Adding member company to an inactive shell.	The partner company {name} cannot be added to inactive shell {name}
902	Partner company specified does not exist as shortname specified is not matching.	The partner company does not exist.
500	Server Error.	Server Error, contact the system administrator.
901	Invalid data value.	Invalid data for <data_element>.
5555	One or more records contain errors.	The partner company does not exist

Remove (Update) Partner Company Shell Membership

Method: PUT

URL: <unifier_url>/ws/rest/service/v1/admin/company/shell/member/remove

Objective: Remove partner company from shell based on information provided.

Usage:

This will be a PUT request.

The service URL to be used will be: `http://<environment url>/ws/rest/service/v1/admin/company/shell/member/remove`

Input:

Company information in JSON format.

Sample JSON to add member company to a shell

```
{
     "shortname":"partner01",
     "shellnumber":"7896"
 }
```

Output:

Return data will be status and message.

Validations:

Status	Condition	Message
200	Success	The member company {name} has been removed from the shell {name}
902	Partner company specified does not exist as shortname specified is not matching.	The partner company does not exist.
602	When shell number is invalid or does not exist.	The shell number is invalid/does not exist.
913	When shell number specified is for inactive shell.	The shell specified is inactive.
500	Server Error.	Server Error, contact the system administrator.
5555	One or more records contain errors.	The partner company does not exist.

User Services V1

Get User

Method: POST

This will not return Hosting and Partner company users (Same as SOAP web service). The Get User REST is different than the SOAP web service. The REST will return the owner company and partner company users, based on the filter condition.

URL: `<unifier_url>/ws/rest/service/v1/admin/user/get`

Input

For filter condition:

```
{
    "filterCondition":"uuu_user_status=1"
}
```

For Get User call, the input filter condition has to be specified.

Create User

Method: POST

Objective: Create partner/company user based on information provided.

Usage:

This will be POST request.

The service URL to be used will be: `http://<environment url>/ws/rest/service/v1/admin/user`

Input:

The user will be required to enter the company/partner user information.

The JSON input to create owner company or partner company users are similar. The Minimum input is:

```
{
    "copyFromUserPreferenceTemplate": "",
    "users": [
        {
            "uuu_user_loginname": "user010",
            "uuu_user_password": "123123",
            "uuu_user_firstname": "User",
            "uuu_user_lastname": "010",
            "uuu_user_email": "user010@oracle.com",
            "uuu_user_type": "0",
            "uuu_user_status": "1",
            "uuu_user_dateformat": "0",
            "uuu_user_timezone_pref": "50",
            "uuu_user_company": "unifier"
        }
    ]
}
```

Sample JSON to create Partner user

```
{
  "copyfromuserpreferencetemplate":"92Template1",
  "users":
  [
    {
      "uuu_user_workphone":"1-800-333-8989",
      "uuu_user_name":"Company Administrator",
      "uuu_user_status":"1",
      "uuu_user_email":"ashish.c.kumar@oracle.com",
      "empEmployeeIDTB50":"",
      "uuu_user_dateformat":"0",
      "uuu_user_proxy":"",
      "uuu_user_type":"0",
      "uuu_user_timezone_pref":"50",
      "uuu_user_pager":"",
      "empEmployeeRegHrsDA":"0.0",
      "uuu_user_homephone":"",
      "uuu_user_fax":"",
      "uuu_user_lastname":"Administrator",
```

```
        "ugenFlrTB120":"",
        "uemSpace1DPK":"",
        "uuu_user_title":"Administrator",
        "uempCurrLocBldDPK":"",
        "uplRoomTB250":"",
        "empEmployeeManagerTB120":"",
        "uuu_user_mobilephone":"1-650-323-32324",
        "uuu_user_proxy_config":"1",
        "usmUSAssignedDeptPD":"",
        "uSiteDP":"",
        "uuu_user_loginname":"coadmin",
        "uuu_user_company":"unifier",
        "uuu_user_firstname":"Company",
        "uuu_user_password" : "123"
      }
    ]
  }
```

For creating company user the minimum requirement will be:

Sample JSON to create Company User

```
  {
  "user": {
          "loginname":"abcuser",
          "password":"123",
          "firstname":"ABC",
          "lastname":"User",
          "mobilephone":"951-109-8347",
          "homephone":"951-109-8347",
          "workphone":"951-109-8347",
          "email":"abc@oracle.com",
          "pager":"951-109-8347",
          "fax":"951-109-8347",
          "title":"Lead",
          "date_format":"951-109-8347",
          "timezone":"951-109-8347",
          "user_type":"Standard"
  },
  }
```

Output:

If the service runs successfully the output is status and message.

The mandatory existing validations that exist for creating partner/company user will be applicable in REST service as well.

Validations for creating partner company

Status	Condition	Message
200	Success	The partner user has been created.
500	Server Error.	Server Error, contact the system administrator.
5555	One or more records contain errors.	One or more records contain errors.

Update User

Method: PUT

Objective: Update company/partner user based on information provided.

> **Note**: In order to update the user successfully, ensure that the integration user has the proper update permissions (Company Workspace (Admin mode) > User Administration > Integration Users).

Usage:

This will be a PUT request.

The service URL to be used will be: `http://<environment url>/ws/rest/service/v1/admin/user`

Input:

User information in JSON format

Sample JSON to update Partner user

```
{
 "users":
  [
    {
      "uuu_user_workphone":"1-800-333-8989",
      "uuu_user_name":"Company Administrator",
      "uuu_user_status":"1",
      "uuu_user_email":"ashish.c.kumar@oracle.com",
      "empEmployeeIDTB50":"",
      "uuu_user_dateformat":"0",
      "uuu_user_proxy":"",
      "uuu_user_type":"0",
      "uuu_user_timezone_pref":"50",
```

```
        "uuu_user_pager":"",
        "empEmployeeRegHrsDA":"0.0",
        "uuu_user_homephone":"",
        "uuu_user_fax":"",
        "uuu_user_lastname":"Administrator",
        "ugenFlrTB120":"",
        "uemSpace1DPK":"",
        "uuu_user_title":"Administrator",
        "uempCurrLocBldDPK":"",
        "uplRoomTB250":"",
        "empEmployeeManagerTB120":"",
        "uuu_user_mobilephone":"1-650-323-32324",
        "uuu_user_proxy_config":"1",
        "usmUSAssignedDeptPD":"",
        "uSiteDP":"",
        "uuu_user_loginname":"coadmin",
        "uuu_user_company":"unifier",
        "uuu_user_firstname":"Company"
        "uuu_user_password" : "123"
      }
    ]
  }
```

Output:

Return data will be JSON format.

- `uuu_user_loginname` will be the identifier to identify if the user can be updated
- `uuu_user_company` will be `URLencoded`

Validations for updating company partner user

Status	Condition	Message
200	Success	The {first name last name} user has been updated.
500	Server Error.	Server Error, contact the system administrator.
5555	One or more records contain errors.	One or more records contain errors.

Update User Shell Membership

Method: PUT

URL: <unifier_url>/ws/rest/service/v1/admin/user/shell/membership

Sample input JSON

```
{
    "shellnumber":"State0003",
    "users":[
        { "username":"ConfUser3",
          "status":"Active",
          “group_add”: “G1”,
                  “group_remove”: “Project Administrator”
        }
    ]
}
```

Update User Group Membership

Method: PUT

URL: <unifier_url>/ws/rest/service/v1/admin/user/group/membership

Sample input JSON

```
[
  {
      "username":"ConfUser9",
       "group_add":"Group_A",
       "group_remove":"new group"
  }
]
```

User Defined Report (UDR) Services V1

Usage:

POST /ws/rest/service/v1/data/udr/get/{project_number}

Purpose:

Get User Defined Report Data in a project/shell based on {project_number} or from Company level if {project_number} is not provided.

Input:

Path Parameter

project_number: Specify the Project/shell number to generate the UDR data, if not provided then UDR will be generated from Company Level.

UDR information in JSON format

Minimum data requirement is:

```
{
"reportname":"partner01_UDR"
}
```

Method : POST

With project Number (Shell level reports): http://<environment url>/ws/rest/service/v1/data/udr/get /<project-number>

Method : POST

Input:

UDR information in JSON format

Minimum data requirement is:

```
{
"reportname":"partner01_UDR"
}
```

Output:

Return data will be JSON format by default.

Validations:

Error Code	Condition	Message
200	Success	Ok
500	Server Error	Server Error. Contact the system administrator.
403	Validate the query exists or not.	The query condition {0} does not exist.
403	Validate values mentioned correct or not (Does it require value 2 for between condition?)	Invalid Data. (Value2 is required only when a 'between' query condition exists for the query.
763	Duplicate query labels	Duplicate query label found.

Passing Values to UDR

To pass parameter for query condition for a UDR:

Sample JSON to pass parameters

```
{
"reportname": "UDR for Request for Bid",
"query": [
{
"label":"Requests for Bid / Title",
"value1":"Standard"
},
{
"label":"Requests for Bid / Title",
"value1":"34",
"value2":"43"
},
{
"label":"Formula_01",
"value1":"100"
},
{
"label":"Formula_02",
"value1":"200"
},
{
"label":"Requests for Bid / Due Date",
"value1":"2018-07-31",
"value2":"5"
},
{
"label":"Requests for Bid / Bid Sponsor",
"value1":"XYZ"
}
]
}
```

Parameter for passing values:

Parameter	Condition
label	This must match the query condition label in the UDR.
Value 1	This will help user to enter value for given query condition.
Value 2	This parameter will help user to enter the secondary value in case of between query condition.

Get Templates List

POST /ws/rest/service/v1/cashflow/template/list/{project_number}

Purpose:

Get all list of CashFlow Templates with all curves types defined.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: (Required) Specify the Project number in which the curve exists.

POST body is a JSON

Note: POST call has input & output both as JSON in the body.

```
{
"options":{
"property_name": "CBS CF"
}
}
```

Supported options properties

"property_name": "CBS CF"

"curve_name": << Name of The Curve>>,

"rollup_status"<<Active><Inactive>>

If rollup_status value provided is other than above mentioned values, invalid options message will be thrown.

"detail_level"<<Summary CBS>< CBS><Project / Shell><Commitment>>

If detail_levelvalue provided is other than above mentioned values,invalid options message will be thrown.

"include_curves":<< Yes/ No>> <!-- default is Yes if not provided→>

If include_curves value provided is other than above mentioned values, invalid options message will be thrown.

If no options provided service will return all templates along with cashflow curves from the project.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Project or Shell Sample


```
{
    "data": [
        {
            "name": "Project Cash Flow - Capital Planning",
            "description": null,
            "rollup_status": "Active",
            "detail_level": "Project / Shell",
            "period_type": "Standard Planning Period",
            "period_name": "Standard Planning Period",
            "period_by": "Month",
            "period_format": "M YYYY",
            "decimal_places": 2,
            "curves": [
                {
                    "name": "Baseline",
                    "type": "Baseline",
                    "distribution": {
                        "type": "auto_profile",
                        "distribution_profile": "S Curve"
                    },
                    "cost": {
                        "type": "cost_sheet",
                        "cost_sheet_column": "Revised Budget"
                    },
                    "schedule": {
                        "type": "manual"
                    }
                },
                {
                    "name": "Actuals",
```

```
            "type": "Spends",
            "cost": {
                "cost_sheet_column": "Spends",
                "type": "cost_sheet"
            }
        },
        {
            "name": "Forecast",
            "type": "Forecast",
            "distribution": {
                "type": "auto_profile",
                "distribution_profile": "S Curve"
            },
            "cost": {
                "type": "cost_sheet",
                "cost_sheet_column": "Forecast"
            },
            "schedule": {
                "type": "manual"
            },
            "forecast_options": {}
        },
        {
            "name": "Original Budget",
            "type": "Portfolio Budget",
            "cost": {
                "currency_code": "United States Dollar (USD)",
                "distribut_amount_from": "original"
            }
        },
        {
            "name": "Approved Budget",
            "type": "Portfolio Budget",
            "cost": {
```

```
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "approved"
                    }
                },
                {
                    "name": "Shared Budget",
                    "type": "Portfolio Budget",
                    "cost": {
                        "currency_code": "United States Dollar (USD)",
                        "distribut_amount_from": "shared"
                    }
                }
            ],
            "filters": {
                "filter_option": "all"
            }
    ],
    "message": [
        "success"
    ],
    "status": 200
}
```


Data

If integration interface is defined for BP, then integration form will be used for this service; otherwise, all custom DEs defined in the form will be used.

For update, specify the DEs to be updated, only.

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.

Multi-part/form-data will be used to handle files.

REST API Details in Event Driven Notification

In order to integrate Unifier with other third party software seamlessly, Unifier will store the relevant events in the database and provide an integration end point making those events available to the external program.

All of the information that is captured within the **Event Notifications** log, about each event, will be be available by way of a REST API: 'GET' (GET /ws/rest/service/v1/event/notification). This API will help you to get all of the notification events by using the available filter parameters.

The following data can be pulled about any event:

#	Attribute name/label in Unifier	JSON label
1	project_id	project_id
2	object_prefix	object_prefix
3	Shell Number	shell_number
4	Shell Name	shell_name
5	Object Type	object_type
6	Object Sub-Type	object_subtype
7	Object Name	object_name
8	Record Number	record_no
9	Event Date	event_date
10	WF Step From WF type BPs only	workflow_from
11	WF Step To WF type BPs only	workflow_to
12	WF Action Name WF type BPs only	workflow_action
13	Old Status	old_status
14	New Status	new_status

In This Section

Get Event Driven Notification..... 533

Get Event Driven Notification

GET /ws/rest/service/v1/event/notification

Purpose:

Get all Notification events based on user configuration/request

Input:

All parameters are optional.

Filter condition example:

"filter":{event_date:"2015-07-04T12:08:56.235",object_type:"Business Process", shell_number:"XTB-100"}

Available filter parameters→ "object_type", "event_date", "object_name", "old_status","new_status", "record_no", "project_id" & "shell_number".

Supported Event Date Format: "2019-11-13T10:42" & with milliseconds "2019-11-13T10:42:19.799"

"max_records": Optional(default 1000 records) -maximum records in the request, if not provided system will return 1000 records.

JSON object containing 'status', 'data', 'message', 'latest_event_date', 'total_records', 'fetched_records'

A message will be present if the status is not 200 otherwise it will be "success".

Sample Request/Output:


```
{
"data":
     {
     "latest_event_date": "2019-12-05T16:01:04.062",
     "total_records": 2,
     "items": [
               {
                    "workflow_from": "Review",
                    "workflow_action": "To Approver",
                    "workflow_to": "Approve",
                    "object_type": "Business Process",
                    "shell_number": "XTB-100",
                    "object_prefix": "uxsov",
                    "project_id": 1002,
                    "object_name": "SOV-W-100",
                    "record_no": "uxsov-0031",
                    "event_date": "2019-12-05T11:16:26.918",
                    "object_subtype": "WorkFlow",
```

```
                    "old_status": "Rejected",
                    "new_status": "Pending"
               },
               {
                    "workflow_from": null,
                    "workflow_action": null,
                    "workflow_to": null,
                    "object_type": "Business Process",
                    "shell_number": "XTB-100",
                    "object_prefix": "uxnwf1",
                    "project_id": 1002,
                    "object_name": "Commit NWF1",
                    "record_no": "uxnwf1-0004",
                    "event_date": "2019-12-04T08:47:46.994",
                    "object_subtype": null,
                    "old_status": "Pending",
                    "new_status": "Pending"
               }
     ],
     "fetched_records": 2
},
"message": [ "Success"],
"status": 200
}
```

REST Web Services V2

HTTP Methods

GET, POST, PUT,DELETE

Authentication

Pre-requisite for invoking any rest service - Integration user and JWT auth token should be available.

Integration User:

All rest services should be accessed via Integration User.

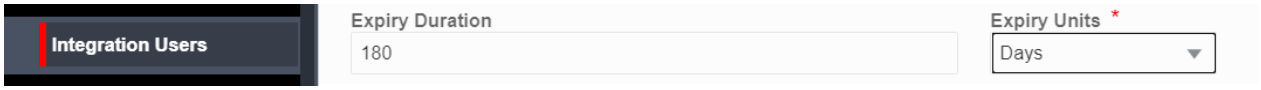
Create an Integration User at Company Workspace → User Administration → Integration User.

Make sure the integration user has the required module permissions enabled.

Integration user token expiration can be set from the Integration user UI. User can create, update Auth Token expiry date to - days, hours and minutes.

1 to 500 is a valid value for Expiry Duration.

By default the value is set to 180 days.



Auth Token:

authentication token is required to access the rest services.

Token is set to be valid for date available in the response.

The token can be obtained by the below url.

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

Method - GET

Headers -Authorization:Basic <Basic Auth of integration user>

(To Generate one via Postman rest client - Go to 'Authorization' tab → choose 'Basic Auth' in 'Type' dropdown → Enter Integration user name and password -. Click on 'Update Request'. This generates the necessary Authorization header in 'Headers' tab.)

If the user name/password combination shared in the above Authorization Header is not valid/correct, the service throws **401** status code(Unauthorized)

Other conditions that are checked are - Integration user has to be active.

The expiry date is displayed as per the user date format preferences set in the Integration UI.

Sample response -

```
 {
"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-F93
1-3F14-C6FA7576F55E7D8D9975C46B5805179BD10D890DF15F"
}
```

Token can now be used in all rest services for authentication mechanism. All rest services should have the below set in the 'Header'.

Header - Sample

Key	Value
Authorization	Bearer eyJ0eXAiOiJEQiJ9.eyJ1c2VybmFtZSI6IiQkZGVsdDMifQ==.02318C44-9F3A-F931-3F14-C6FA7576F55E7D8D9975C46B5805179BD10D890DF15F

Notes:

- Validity of the token is as given in the login. The same can be re-used for subsequent rest requests until the expiry date.
- User can change Expiry Date to minutes, hours and days in UI. Rest will honor those settings and generate new token with given Expiry Duration in the UI.
- If the Authorization token is not valid/correct for subsequent rest requests, those services will throw **401** status code(Unauthorized)
- If the Authorization token is correct but user (for whom token is generated) does not have permission for any rest request, that service will throw **403** status code(Forbidden).
- For every login rest service initiated, a new token is generated, old token is invalidated.

Data

If integration interface is defined for BP then integration form will be used for this service, otherwise all custom DEs defined in the form will be used.

For update, specify only the DEs to be updated.

Data Format

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

Data Transfer

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

Standards

Get Method -

Will be used to request data from a source when no parameters are sent in the body.

> **Note:** Do not use word "get" in the url.

URL Encoding GET call parameter values

All parameters in GET call must be URL encoded.

For Postman REST client,Use below code in "**pre-request Script**" tab, that will trim extra spaces in params key and encode special characters in params value.

```
pm.request.url.query.all().forEach( (param) =>
{
        param.key = param.key.trim();
        param.value = encodeURIComponent(param.value );
    }
);
```

POST Method -

POST can be used to retrieve data with parameters in body.

Can be used for CREATE, Retrieve data with parameters.

For example getBprecordlist - bpname is mandatory and need to be a sent in request body.

> **Note:** Do not use word "create" in the url.

To distinguish create and get data with parameters term list can be added in the url.

PUT Method -

PUT is used update data.

> **Note:** Do not use word "update" in the url.

Logging:

All REST operations on Unifier gets audited in internal audit log table (not accessible to User, as it is not business case audit logs) .

A background CRON job is created to run on every SUNDAY 4:00AM (server time zone) which will purge older REST internal audit logs which goes beyond 25000 audit rows.

IP Filtering:

Customers have the ability to provide the list of IP addresses which can consumer Unifier REST Webservices(V1 or V2). IP Filtering option is available in Unifier portal in Company Properties - Security Tab.

If the 'IP Filtering Policy' field is checked in company properties, then the remote host will be validated based on their IP Version

1. If the remote host is IPv4 version then it will be checked against the list of IP addresses provided in the IPv4 text box.
2. If the remote host is of IPv6 version it will be checked against the list of IPs provided in the IPV6 text box.

3. If IP addresses are provided in CIDR format then the remote host IP will be checked against all addresses that come in the range.

In This Section

REST API Details in Business Processes 540

REST API Details in Business Processes

Create BP Record

POST /ws/rest/service/v2/bp/record

Purpose:

Create a record in a specific BP in a shell based on "project_number" (provided in input JSON options) or from Company level if the project or shell number is not provided.

The input JSON provides various options to be considered for fetching the data.

This V2 service will use the step form (creation form) to create the BP record. All of the required fields and validation rules will be run against the creation form used in the workflow details.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body.

Create BP Record Input JSON

```
{
   "options":{
      "project_number" : "P-0002",
      "bpname":"Vendors",
      "workflow_details":{
         "workflow_name":"workflow_name",
         "user_name":"first_name  last_name",
         "action_name":"action_name"
      }
   },
   "data":[
      {
         "record_no":"hello",
         "uuu_user_id":"a@abc.com",
         "title":"v-00101",
```

```
        "uveLicenseNoTB16":null,
        "uuu_contact_company":"v-00101",
        "uveCOIAmoutCA":0,
        "_bp_lineitems":[
           {
              "uirCntctFstNmTB":"a",
              "uuu_tab_id":"List of Contacts",
              "title":"t",
              "short_desc":"Vendor Contact",
              "uriCntctLstNmTB":"b",
              "ugenStatePD":null,
              "ugenZipCodeTXT16":null,
              "uveEmailTB120":"hello@abc.com",
              "ugenAddress3TXT120":null
           }
        ],
        "creator_id":"Company Administrator",
        "uveVendorIDTB16":"v-00101",
        "uuu_contact_first_name":"a",
        "status":"Active"
     }
  ]
}
```


Notes:

- If "record_no" is not provided, then an auto-generated "record_no" will be assigned to the record. If a unique "record_no" is provided, then the record will be created with a record number.
- Field properties such as Read-only in step form (creation form) will be honored. The Field mandatory in step form must be present in Integration Form with direction set as "both" or "input"
- Field direction provided in Integration form will be honored.
- Auto-Creation setting is mandatory in Shell or Company BP Setup in the Admin mode.
- In input JSON options, the "project_number" specifies the shell in which the records exist, if the "project_number" is not provided, then the records are considered to be from Company Level.
- The "bpname" is mandatory and is part of options form parameter.

- If the "workflow_details" is not specified, then the the auto-creation settings, defined on the BP setup, will be used.
- If the "workflow_details" are specified, then the three params "user_name," "workflow_name," and "action_name" are mandatory.
- The validations will be performed:
  - If the user is a valid active user in the project.
  - The workflow name is valid and active.
  - User is an assignee (user/group) on the creation step of the workflow.
  - Action name is valid outgoing link from the creation step.
  - The following error messages may display:
    - Enter a valid or an active user. (Missing key or invalid user name)
    - The workflow name is not valid. (Missing key or invalid/inactive workflow name)
    - The user does not have an active workflow template. (User is not an assignee on the creation step)
    - The workflow action name is not valid. (Missing key or invalid outgoing action name from creation step)

Output:

JSON object containing 'status', 'data', 'message'

Sample: Create BP Response


```
{
    "data": [],
    "message": [
        {
                "_record_status": "success",
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
                "_bp_lineitems": [
                    {
                        "uirCntctFstNmTB": "a",
                        "uuu_user_workphone": null,
```

```
            "uuu_tab_id": "List of Contacts",
            "title": "t",
            "ugenAddress1TXT120": null,
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "ugenCityTXT50": null,
            "short_desc": "Vendor Contact",
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null
        }
    ],
    "uvePolicyNoTB32": null,
    "record_no": "hello",
    "_attachment": [
        {
            "file_name": "new 1.txt"
        }
    ],
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
    "ugenZipCodeTXT16": null,
    "uveEmailTB120": "a@abc.com",
    "ugenAddress3TXT120": null,
    "uveReferenceIdTB16": null,
    "uuu_creation_date": "04-12-2018",
    "ugenRemarksTB4000": null,
    "uvePhoneTB64": null,
    "uveTaxIDTB16": null,
```

```
                "uveMinorityBusCB": 0,
                "uveInsuranceCoTB32": null,
                "ugenExpirationDateDOP": null,
                "uveWomanOwnedCB": 0,
                "uveVendorNameTB50": "v-00101",
                "ugenAddress1TXT120": null,
                "uveDisadvantagedBusCB": 0,
                "ugenDiscipline": null,
                "creator_id": "Company Administrator",
                "ugenAddress2TXT120": null,
                "ugenCountryPD": null,
                "uveVendorIDTB16": "v-00101",
                "ugenStatePD": null,
                "uuu_contact_first_name": "a",
                "status": "Active"
            }
    ],
    "status": 200
}
```


in case of any error:


```
   {
       "data": [],
       "message": [
           {
               "_record_status": "Error Business Process record_no hello
   already exists. ",
               "record": {
                   "uuu_user_id": "a@abc.com",
                   "uuu_record_last_update_date": "04-12-2018",
                   "uveFaxTB16": null,
                   "uveCertificateNoTB64": null,
                   "uvePrimaryContactTB64": "a",
                   "title": "v-00101",
                   "uveLicenseNoTB16": null,
                   "uuu_contact_company": "v-00101",
                   "uveCOIAmoutCA": 0,
```

```
"_bp_lineitems": [
    {
        "uirCntctFstNmTB": "a",
        "uuu_user_workphone": null,
        "uuu_tab_id": "List of Contacts",
        "title": "t",
        "ugenAddress1TXT120": null,
        "ugenAddress2TXT120": null,
        "ugenCountryPD": null,
        "ugenCityTXT50": null,
        "short_desc": "Vendor Contact",
        "uriCntctLstNmTB": "b",
        "ugenStatePD": null,
        "ugenZipCodeTXT16": null,
        "uveEmailTB120": "hello@abc.com",
        "ugenAddress3TXT120": null
    }
],
"uvePolicyNoTB32": null,
"record_no": "hello",
"_attachment": [
    {
        "file_name": "new 1.txt"
    }
],
"uveVendorTypePD": "Architect",
"ugenCityTXT50": null,
"uveCoiExpDOP": null,
"uuu_dm_publish_path": "v_path",
"uuu_contact_last_name": "b",
"ugenZipCodeTXT16": null,
"uveEmailTB120": "a@abc.com",
"ugenAddress3TXT120": null,
"uveReferenceIdTB16": null,
"uuu_creation_date": "04-12-2018",
"ugenRemarksTB4000": null,
"uvePhoneTB64": null,
"uveTaxIDTB16": null,
"uveMinorityBusCB": 0,
"uveInsuranceCoTB32": null,
```

```
                    "ugenExpirationDateDOP": null,
                    "uveWomanOwnedCB": 0,
                    "uveVendorNameTB50": "v-00101",
                    "ugenAddress1TXT120": null,
                    "uveDisadvantagedBusCB": 0,
                    "ugenDiscipline": null,
                    "creator_id": "Company Administrator",
                    "ugenAddress2TXT120": null,
                    "ugenCountryPD": null,
                    "uveVendorIDTB16": "v-00101",
                    "ugenStatePD": null,
                    "uuu_contact_first_name": "a",
                    "status": "Active"
                }
            }
        ],
        "status": 3000
    }
```


Status codes are:

1>200, for success.

2>3000, for Partial success.

Update BP Record

PUT /ws/rest/service/v2/bp/record

Purpose:

Update a record in a specific BP in a shell based on "project_number" (provided in input JSON options) or from Company level if the project or shell number is not provided.

The input json shall provide various options to be considered for fetching the data.

This V2 service will use the step form to update the BP record. All of the required fields and Validation rules will be run against the step form used in the workflow details.

Form in the End step is a View Form and cannot be updated.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body.

Update BP Record Input


```
{
   "options":{
```

```
      "project_number" : "P-0002",
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS",
      "optimizedPickerExecution" : "true",
      "workflow_details":{
         "WFCurrentStepName":"Step 4",
         "WFActionName":"Line 3"
      }
   },
   "data":[
      {
         "ugenUserIDPK":null,
         "_bp_lineitems":[
            {
               "ugenShellPK":"/AP1",
               "uuu_tab_id":"Standard",
               "short_desc":"desc3_m11",
               "upwoLISAS":"0001",
               "uuu_line_item_status":"Active"
            }
         ],
         "record_no":"uxli6-0020",
         "title":"new title"
      }
   ]
}
```


Notes:

- Field properties such as Read-only in step form will be honored. The Field mandatory in step form must be present in the Integration Form with direction set as "both" or "input".
- The Field direction provided in the Integration form will be honored.
- Auto-Creation setting is mandatory in Shell or Company BP Setup in Admin mode.
- In input JSON options, the "project_number" specifies the shell in which the records exists, if the "project_number" is not provided, then the records are considered to be from Company Level.
- The "record_no" is mandatory and is part of data.

- The "bpname" is mandatory and is part of options form parameter.
- If a Workflow BP record is already accepted by an assignee, then the Update REST call will fail with error message: Record has already been accepted by Task Assignee.
- The WFCurrentStepName and WFActionName are part of the "workflow_details" and are for Workflow-type BPs. If the values for the WFCurrentStepName and WFActionName are provided, then the record will be updated and sent to the next step of the workflow. If the values for the WFCurrentStepName and WFActionName are not provided, then the record will be updated but will remain in same workflow step.
- The LineItemIdentifier is a part of options form parameter. The LineItemIdentifier value will be the DE name present in the lineItem. The DE name present in the lineItem should be present in the "data" (input record JSON) with its value.
- The optimizedPickerExecution is a part of the options form parameter. The optimizedPickerExecution controls the pickers 'populate' execution, on the existing line items. The values can be True or False.

  When the value is True, the Upper Form, or the Lineitem Form, input JSON is required to send the Datapicker DE tag (with values) to trigger 'populate' on the other DEs that are configured per design. If the input JSON does not contain the Datapicker DE property, the system will not process the Datapicker DE as a part of the update to 'populate' the values from the data picker record to the current record and lineitems that get updated.

  When the value is False, or non-existent, the system will continue to process according to the existing behavior.

Steps to add LineItemIdentifier in the BP design:

1. Create a custom DE whose Data definition is 'Sys Auto Sequence'.
2. Add this DE to the detail form of the BP whose line items needs to be updated.
3. Navigate to Company Workspace > Business Processes and open the BP.
4. Navigate to the BP-Home > click Open drop-down > Data Elements. A data element configuration screen appears
5. Click on Auto Sequence tab and click Add.
6. Select the Data element added in the detail form and select the level as 'Per record'.
7. Provide the start value and click Create (next to format), add the parameter 'Sys Sequence counter', and click OK.
8. Add the DE to Integration > Detail and select "Direction" to "Both" for the DE.
9. Complete and deploy the BP.
10. In the body of the request add this DE name as Lineitem identifier and send the request.

Delete the Line-Item, using Update Request:

To delete the line-Item using update request, the field "_delete_line_item" should be used (value of which will be comma separated values of LineItemIdentifier DE) as shown in the following example:

Sample: Delete line-item input

```
{
   "options":{
      "project_number" : "P-0002",
```

```
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS"
   },
   "data":[
      {
         "ugenUserIDPK":null,
         "_delete_bp_lineitems":"0002,0001,0003",
         "record_no":"uxli6-0021",
         "title":"title_m112244"
      }
   ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 .

Status codes are:

1>200 OK, if all update succeeds.
2>3000, if any of records update fails.

Sample: Update BP Record Response

```
{
    "data": [],
    "message": [
        {
            "_record_status": "success",
            "record": {
                "ugenUserIDPK": null,
                "record_no": "uxli6-0021",
                "title": "title_m112244"
            }
        }
    ],
    "status": 200
}
```

Data Elements in Line Items

If the following Data Elements are present in the Line Items, they will not be updated during UpdateRecordV2 call.

- bitemid
- budgetid
- sovitemid
- uuu_cm0_picker
- xid
- ds_code
- fsm_lineitem_id
- asm_lineitem_id
- uuu_lat
- uuu_long
- uuu_company_account_name
- uuu_company_account_code
- uuu_company_acc_codepicker
- uuu_activity_picker
- uuu_asset_picker
- ref_bpo_lineitem
- ref_bpo
- ref_rfb
- refid
- uuu_sovlinum
- otherCompanyID
- uuu_costcode_picker
- scheduled_value
- uuu_asset_cum_depreciation
- uuu_asset_curr_period_dep
- uuu_asset_net_book_value
- uuu_unit_price
- currencyid
- due_date
- uuu_asset_calc_as_of_date
- uuu_from_date
- uuu_issue_date
- uuu_last_update_date
- uuu_rfb_due_date
- uuu_to_date
- uuu_week_picker
- uuu_datasource
- uuu_depreciation_mtd
- uuu_line_item_status

- row_id
- uuu_bid_count
- uuu_bidders_count
- uuu_planning_item_picker
- uuu_proj_picker
- record_no
- uuu_resc_picker
- uuu_role_picker
- uuu_commit_short_desc
- end_date
- uuu_creation_date
- uuu_asset_name
- uuu_commit_breakdown
- uuu_asset_code
- uuu_asset_navigation_code
- uuu_timescale_units
- creator_id
- wpid
- uuu_cost_costattribute
- uuu_cost_external_refid
- uuu_cost_status
- uuu_cost_cost_type
- uuu_cost_description
- uuu_cost_item
- uuu_cost_owner
- uuu_cost_code
- uuu_dm_create_date
- uuu_file_create_date
- uuu_file_issue_date
- uuu_dm_percent_complete
- uuu_file_size
- uuu_file_version
- uuu_dm_description
- uuu_dm_node_path
- uuu_dm_node_name
- uuu_file_revision_no
- uuu_file_title
- uuu_dm_create_by
- uuu_file_create_by
- uuu_fund_fundcategory
- uuu_fund_long_desc

- uuu_fund_description
- uuu_fund_fundname
- uuu_fund_code
- uuu_description
- uuu_resc_nwd_type
- uuu_resc_interest
- uuu_resc_proficiency
- uuu_resc_skill
- uuu_role_status
- uuu_role_name
- uuu_resc_capacity
- uuu_user_address
- uuu_user_city
- uuu_user_country
- uuu_user_email
- uuu_user_fax
- uuu_user_firstname
- uuu_user_homephone
- uuu_user_lastname
- uuu_user_mobilephone
- uuu_user_pager
- uuu_user_state
- uuu_user_timezone
- uuu_user_title
- uuu_user_workphone
- uuu_user_zip
- uuu_resc_status
- uuu_resc_code
- uuu_resc_name
- uuu_early_finish
- uuu_early_start,uuu_finish
- uuu_late_finish
- uuu_late_start
- uuu_start
- uuu_act_pct_complete
- uuu_activity_work_hrs
- uuu_duration
- uuu_fixed_cost
- uuu_float
- uuu_labor_cost
- uuu_non_labor_cost

- uuu_total_cost
- uuu_activity_id
- uuu_flag_complete
- uuu_flag_milestone
- uuu_activity_resources
- uuu_dependency_type
- uuu_activity_name
- uuu_outline_code
- uuu_cost_li_type

Update BP Record with Attachment (REST API Details in Business Processes)

PUT /ws/rest/service/v2/bp/record/file

Purpose:

Update a record with attachment in a specific BP in a shell based on "project_number" (provided in input JSON options) or from Company level if the project or shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

This V2 service will use the step form to update the BP record. All of the required fields and Validation rules will be run against the step form used in the workflow details.

Form in the End step is a View Form and cannot be updated.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body.

Update BP Record with Attachment

```
{
   "options":{
      "project_number" : "P-0002",
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS",
      "workflow_details":{
         "WFCurrentStepName":"Step 4",
         "WFActionName":"Line 3"
      }
   },
   "data":[
      {
         "ugenUserIDPK":null,
```

```
         "_bp_lineitems":[
            {
               "ugenShellPK":"/AP1",
               "uuu_tab_id":"Standard",
               "short_desc":"desc3_m11",
               "upwoLISAS":"0001",
               "uuu_line_item_status":"Active"
            }
         ],
         "record_no":"uxli6-0020",
         "title":"new title",
         "_attachment":[
            {
               "file_name":"file_1.txt",
               "title":"file_title1",
               "issue_date":"05/06/2018",
               "revision_no":"300"
            },
            {
               "file_name":"file_2.txt",
               "title":"file_title2",
               "issue_date":"05/06/2018",
               "revision_no":"200"
            }
         ]
      }
   ],
   "_attachment":
   {
        "zipped_file_name" : "zip_file_name.zip",
        "zipped_file_size": "746089",
        "zipped_file_content" :"<base64_encoded string of create.zip
file>"
    }
```

}

Notes:

- If "adding attachment" is disabled in Upper form or Detail form, and user is trying to attach file to corresponding form. Updating record will fail.
- Only one record update with attachments is supported in this call. If the file already exists in record, then that file will be ignored and will not get revised .
- If step form has "add attachment" disabled, then files will not get attached.
- If the "_attachment" (outside data list) for zip file is not provided, then the record will be updated without attachment.
- Field properties such as Read-only in step form will be honored. The Field mandatory in step form must be present in Integration Form with the direction set as "both" or "input".
- Field direction provided in Integration form will be honored.
- Auto-Creation setting is mandatory in Shell or Company BP Setup in the Admin mode.
- In input JSON options, the "project_number" specifies the shell in which the records exist, if the "project_number" is not provided, then the records are considered to be from Company Level.
- The "record_no" is mandatory and is a part of data.
- The "bpname" is mandatory and is a part of options form parameter.
- If a Workflow BP record is already accepted by an assignee, then Update REST call will fail with error message: "Record has already been accepted by Task Assignee".
- The WFCurrentStepName and WFActionName are part of the "workflow_details" and are for Workflow type BPs. If the values for WFCurrentStepName and WFActionName are provided, then the record will be updated and sent to next step of the workflow. If the values for WFCurrentStepName and WFActionName are not provided, then the record will be updated but it will remain on the same workflow step.
- The LineItemIdentifier is part of options form parameter. The LineItemIdentifier value will be DE name present in the lineItem. That DE should be present in "data" (input record JSON) with its value.

Steps to add LineItemIdentifier in the BP design:

1. Create a custom DE with DD: Sys Auto Sequence.
2. Add this DE to the Detail Form of the BP with the line items that need to be updated.
3. Navigate to the Company Workspace > Admin mode > Business Processes and open the BP.
4. Click Open drop-down to open and select Data Elements to open the Data Elements Configuration window.
5. Click the Auto Sequence tab and click Add.
6. Select the data element added in the Detail Form and select the level as: Per Record.
7. Provide the start value and click Create.
8. Add the parameter "Sys Sequence counter" and click OK.
9. Add the DE to Integration > Detail.
10. Set the "Direction" to "Both" for that DE.
11. Complete and deploy the BP.

I2. In the body of the request, add the DE name as Lineitem identifier and send the request.

Delete the Line-Item, using Update Request:

To delete the line-Item using update request, the field "_delete_line_item" should be used (value of which will be comma separated values of LineItemIdentifier DE) as shown in the following example:

Sample: *Delete line-item input*

```
{
   "options":{
      "project_number" : "P-0002",
      "bpname":"line6",
      "LineItemIdentifier":"upwoLISAS"
   },
   "data":[
      {
         "ugenUserIDPK":null,
         "_delete_bp_lineitems":"0002,0001,0003",
         "record_no":"uxli6-0021",
         "title":"title_m112244"
      }
   ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 .

Status codes are:

1>200 OK, if all creation succeeds.

2>3000, if any of records creation fails.

Sample: *Update record with attachment response*

```
   {
      "data":[
      ],
      "message":[
         {
            "_record_status":"success",
            "record":{
```

```
            "ugenUserIDPK":null,
            "record_no":"uxli6-0021",
            "title":"title_m112244"
         }
      }
   ],
   "status":200
}
```


or if zip_file_size of uploaded zip file is not correct:


```
{
    "data": [],
    "message": [
        "Uploaded Zip file is invalid"
    ],
    "status": 3003
}
```


Data Elements in Line Items

If the following Data Elements are present in the Line Items, they will not be updated during UpdateRecordV2 call.

- bitemid
- budgetid
- sovitemid
- uuu_cm0_picker
- xid
- ds_code
- fsm_lineitem_id
- asm_lineitem_id
- uuu_lat
- uuu_long
- uuu_company_account_name
- uuu_company_account_code
- uuu_company_acc_codepicker
- uuu_activity_picker
- uuu_asset_picker
- ref_bpo_lineitem
- ref_bpo
- ref_rfb
- refid
- uuu_sovlinum
- otherCompanyID

- uuu_costcode_picker
- scheduled_value
- uuu_asset_cum_depreciation
- uuu_asset_curr_period_dep
- uuu_asset_net_book_value
- uuu_unit_price
- currencyid
- due_date
- uuu_asset_calc_as_of_date
- uuu_from_date
- uuu_issue_date
- uuu_last_update_date
- uuu_rfb_due_date
- uuu_to_date
- uuu_week_picker
- uuu_datasource
- uuu_depreciation_mtd
- uuu_line_item_status
- row_id
- uuu_bid_count
- uuu_bidders_count
- uuu_planning_item_picker
- uuu_proj_picker
- record_no
- uuu_resc_picker
- uuu_role_picker
- uuu_commit_short_desc
- end_date
- uuu_creation_date
- uuu_asset_name
- uuu_commit_breakdown
- uuu_asset_code
- uuu_asset_navigation_code
- uuu_timescale_units
- creator_id
- wpid
- uuu_cost_costattribute
- uuu_cost_external_refid
- uuu_cost_status
- uuu_cost_cost_type
- uuu_cost_description

- uuu_cost_item
- uuu_cost_owner
- uuu_cost_code
- uuu_dm_create_date
- uuu_file_create_date
- uuu_file_issue_date
- uuu_dm_percent_complete
- uuu_file_size
- uuu_file_version
- uuu_dm_description
- uuu_dm_node_path
- uuu_dm_node_name
- uuu_file_revision_no
- uuu_file_title
- uuu_dm_create_by
- uuu_file_create_by
- uuu_fund_fundcategory
- uuu_fund_long_desc
- uuu_fund_description
- uuu_fund_fundname
- uuu_fund_code
- uuu_description
- uuu_resc_nwd_type
- uuu_resc_interest
- uuu_resc_proficiency
- uuu_resc_skill
- uuu_role_status
- uuu_role_name
- uuu_resc_capacity
- uuu_user_address
- uuu_user_city
- uuu_user_country
- uuu_user_email
- uuu_user_fax
- uuu_user_firstname
- uuu_user_homephone
- uuu_user_lastname
- uuu_user_mobilephone
- uuu_user_pager
- uuu_user_state
- uuu_user_timezone

- uuu_user_title
- uuu_user_workphone
- uuu_user_zip
- uuu_resc_status
- uuu_resc_code
- uuu_resc_name
- uuu_early_finish
- uuu_early_start,uuu_finish
- uuu_late_finish
- uuu_late_start
- uuu_start
- uuu_act_pct_complete
- uuu_activity_work_hrs
- uuu_duration
- uuu_fixed_cost
- uuu_float
- uuu_labor_cost
- uuu_non_labor_cost
- uuu_total_cost
- uuu_activity_id
- uuu_flag_complete
- uuu_flag_milestone
- uuu_activity_resources
- uuu_dependency_type
- uuu_activity_name
- uuu_outline_code
- uuu_cost_li_type

Create BP Record with Attachment

POST /ws/rest/service/v2/bp/record/file

Purpose:

Create a record in a specific BP in a shell based on project_number (provided in input JSON options) or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

This V2 service will use the step form (creation form) to create a BP record. All of the required fields and Validation rules will be run against the creation form used in the workflow details.

> **Note**: Only one record with attachments is supported in this call.

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body.

Create BP Record with Attachment

```
{
    "options": {
        "project_number" : "P-0002",
        "bpname": "Vendors",
        "workflow_details":
        {
            "workflow_name": "workflow_name",
            "user_name" : "first_name  last_name",
            "action_name" : "action_name"
        }
    },
    "data": [
            {
                "uuu_user_id": "a@abc.com",
                "uuu_record_last_update_date": "04-12-2018",
                "uveFaxTB16": null,
                "uveCertificateNoTB64": null,
                "uvePrimaryContactTB64": "a",
                "title": "v-00101",
                "uveLicenseNoTB16": null,
                "uuu_contact_company": "v-00101",
                "uveCOIAmoutCA": 0,
                "_bp_lineitems": [
                    {
                        "uirCntctFstNmTB": "a",
                        "uuu_user_workphone": null,
                        "uuu_tab_id": "List of Contacts",
                        "title": "t",
                        "ugenAddress1TXT120": null,
                        "ugenAddress2TXT120": null,
                        "ugenCountryPD": null,
```

```
            "ugenCityTXT50": null,
            "short_desc": "Vendor Contact",
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null,
    "_attachment":[
        {
            "file_name":"new 1.txt",
            "title":"file_title",
            "issue_date":"05/06/2018",
            "revision_no":"100"
        }
        ]
        }
    ],
    "uvePolicyNoTB32": null,
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
    "ugenZipCodeTXT16": null,
    "uveEmailTB120": "a@abc.com",
    "ugenAddress3TXT120": null,
    "uveReferenceIdTB16": null,
    "uuu_creation_date": "04-12-2018",
    "ugenRemarksTB4000": null,
    "uvePhoneTB64": null,
    "uveTaxIDTB16": null,
    "uveMinorityBusCB": 0,
    "uveInsuranceCoTB32": null,
    "ugenExpirationDateDOP": null,
```

```
            "uveWomanOwnedCB": 0,
            "uveVendorNameTB50": "v-00101",
            "ugenAddress1TXT120": null,
            "uveDisadvantagedBusCB": 0,
            "ugenDiscipline": null,
            "creator_id": "Company Administrator",
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "uveVendorIDTB16": "v-00101",
            "ugenStatePD": null,
            "uuu_contact_first_name": "a",
            "status": "Active",
            "_attachment":[
                {
                    "file_name":"new 1.txt",
                    "title":"file_title1",
                    "issue_date":"05/06/2018",
                    "revision_no":"300"
                },
                {
                    "file_name":"exp1.txt",
                    "title":"file_title2",
                    "issue_date":"05/06/2018",
                    "revision_no":"200"
                }

                ]
        }
    ]
,
"_attachment": {
         "zipped_file_name" : "create.zip",
         "zipped_file_size": "746089",
```

```
            "zipped_file_content" :"<base64_encoded string of create.zip
file>"
    }
}
```

Notes:

- If "adding attachment" is disabled in Upper form or Detail form, and user is trying to attach file to corresponding form. Creating record will fail.
- Only one record creation with attachments is supported in this call.
- If the creation form has the "add attachment" disabled, then the files will not get attached.
- If the "_attachment" (outside data list) for zip file is not provided, then the record will be created without attachment.
- Field properties such as Read-only in step form (creation form) will be honored. The Field mandatory in step form must be present in Integration Form with direction set as "both" or "input".
- Field direction provided in Integration form will be honored.
- Auto-Creation setting is mandatory in Shell or Company BP Setup in the Admin mode.
- In input JSON options, the "project_number" specifies the shell in which the records exist, if the "project_number" is not provided, then the records are considered to be from Company Level.
- If the "record_no" is not provided, then an auto-generated "record_no" will be assigned to the record. If a unique "record_no" is provided, then the record will be created with a record number.
  - The "bpname" is mandatory and is part of options form parameter.
  - If the "workflow_details" is not specified, then the the auto-creation settings, defined on the BP setup, will be used.
  - If the "workflow_details" are specified, then the three params "user_name," "workflow_name," and "action_name" are mandatory.
  - The validations will be performed:
    - If the user is a valid active user in the project.
    - The workflow name is valid and active.
    - User is an assignee (user/group) on the creation step of the workflow.
    - Action name is valid outgoing link from the creation step.
    - The following error messages may display:
      - Enter a valid or an active user. (Missing key or invalid user name)
      - The workflow name is not valid. (Missing key or invalid/inactive workflow name)
      - The user does not have an active workflow template. (User is not an assignee on the creation step)
      - The workflow action name is not valid. (Missing key or invalid outgoing action name from creation step)
  - If the "_attachment" (outside data list) for zip file is not provided, then the record will be created without attachment.
  - Only one record input must be provided in Create BP record with attachment.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if status is not 200 .

Status codes are:

1>200 OK, if all creation succeeds.
2>3000, if any of records creation fails.

Sample: Create Record with Attachment Response


```
{
    "data": [],
    "message": [
        {
            "uuu_user_id": "a@abc.com",
            "uuu_record_last_update_date": "04-12-2018",
            "uveFaxTB16": null,
            "uveCertificateNoTB64": null,
            "uvePrimaryContactTB64": "a",
            "title": "v-00101",
            "uveLicenseNoTB16": null,
            "uuu_contact_company": "v-00101",
            "_record_status": "success",
            "uveCOIAmoutCA": 0,
            "_bp_lineitems": [
                {
                    "uirCntctFstNmTB": "a",
                    "uuu_user_workphone": null,
                    "uuu_tab_id": "List of Contacts",
                    "title": "t",
                    "ugenAddress1TXT120": null,
                    "_attachment": [
                        {
                            "issue_date": "05/06/2018",
                            "revision_no": "100",
                            "file_name": "new 1.txt",
                            "title": "file_title"
```

```
                }
            ],
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "ugenCityTXT50": null,
            "short_desc": "Vendor Contact",
            "uriCntctLstNmTB": "b",
            "ugenStatePD": null,
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "hello@abc.com",
            "ugenAddress3TXT120": null
        }
    ],
    "uvePolicyNoTB32": null,
    "_attachment": [
        {
            "issue_date": "05/06/2018",
            "revision_no": "300",
            "file_name": "new 1.txt",
            "title": "file_title1"
        },
        {
            "issue_date": "05/06/2018",
            "revision_no": "200",
            "file_name": "exp1.txt",
            "title": "file_title2"
        }
    ],
    "record_no": "VEN-0086",
    "uveVendorTypePD": "Architect",
    "ugenCityTXT50": null,
    "uveCoiExpDOP": null,
    "uuu_dm_publish_path": "v_path",
    "uuu_contact_last_name": "b",
```

```
            "ugenZipCodeTXT16": null,
            "uveEmailTB120": "a@abc.com",
            "ugenAddress3TXT120": null,
            "uveReferenceIdTB16": null,
            "uuu_creation_date": "04-12-2018",
            "ugenRemarksTB4000": null,
            "uvePhoneTB64": null,
            "uveTaxIDTB16": null,
            "uveMinorityBusCB": 0,
            "uveInsuranceCoTB32": null,
            "ugenExpirationDateDOP": null,
            "uveWomanOwnedCB": 0,
            "uveVendorNameTB50": "v-00101",
            "ugenAddress1TXT120": null,
            "uveDisadvantagedBusCB": 0,
            "ugenDiscipline": null,
            "creator_id": "Company Administrator",
            "ugenAddress2TXT120": null,
            "ugenCountryPD": null,
            "uveVendorIDTB16": "v-00101",
            "ugenStatePD": null,
            "uuu_contact_first_name": "a",
            "status": "Active"
        }
    ],
    "status": 200
}
```


In case of any errors:


```
   {
       "data": [],
       "message": [
           {
               "_record_status": "Error Business Process record_no hello
   already exists. ",
               "record": {
```

```
"uuu_user_id": "a@abc.com",
"uuu_record_last_update_date": "04-12-2018",
"uveFaxTB16": null,
"uveCertificateNoTB64": null,
"uvePrimaryContactTB64": "a",
"title": "v-00101",
"uveLicenseNoTB16": null,
"uuu_contact_company": "v-00101",
"uveCOIAmoutCA": 0,
"_bp_lineitems": [
    {
        "uirCntctFstNmTB": "a",
        "uuu_user_workphone": null,
        "uuu_tab_id": "List of Contacts",
        "title": "t",
        "ugenAddress1TXT120": null,
        "ugenAddress2TXT120": null,
        "ugenCountryPD": null,
        "ugenCityTXT50": null,
        "short_desc": "Vendor Contact",
        "uriCntctLstNmTB": "b",
        "ugenStatePD": null,
        "ugenZipCodeTXT16": null,
        "uveEmailTB120": "hello@abc.com",
        "ugenAddress3TXT120": null
    }
],
"uvePolicyNoTB32": null,
"record_no": "hello",
"_attachment": [
    {
        "file_name": "new 1.txt"
    }
],
"uveVendorTypePD": "Architect",
"ugenCityTXT50": null,
"uveCoiExpDOP": null,
"uuu_dm_publish_path": "v_path",
"uuu_contact_last_name": "b",
"ugenZipCodeTXT16": null,
```

```
                    "uveEmailTB120": "a@abc.com",
                    "ugenAddress3TXT120": null,
                    "uveReferenceIdTB16": null,
                    "uuu_creation_date": "04-12-2018",
                    "ugenRemarksTB4000": null,
                    "uvePhoneTB64": null,
                    "uveTaxIDTB16": null,
                    "uveMinorityBusCB": 0,
                    "uveInsuranceCoTB32": null,
                    "ugenExpirationDateDOP": null,
                    "uveWomanOwnedCB": 0,
                    "uveVendorNameTB50": "v-00101",
                    "ugenAddress1TXT120": null,
                    "uveDisadvantagedBusCB": 0,
                    "ugenDiscipline": null,
                    "creator_id": "Company Administrator",
                    "ugenAddress2TXT120": null,
                    "ugenCountryPD": null,
                    "uveVendorIDTB16": "v-00101",
                    "ugenStatePD": null,
                    "uuu_contact_first_name": "a",
                    "status": "Active"
                }
            }
        ],
        "status": 3000
    }
```


or if zip_file_size of uploaded zip file is not correct:


```
    {
        "data": [],
        "message": [
            "Uploaded Zip file is invalid"
        ],
        "status": 3003
    }
```

Response Error Codes (REST API Details in Business Processes)

Code	Message	Details
200	success	If the request is successful
500	Server Error, contact the system administrator.	If there is any exception when running this service
603	Business Process Name is not correct.	
650	Create Business Process failed reasons, files cannot be attached to upper form. Create Business Process failed reasons, files cannot be attached to detail form: <tab_name> Update Business Process failed reasons, files cannot be attached to upper form. Update Business Process failed reasons, files cannot be attached to detail form: <tab_name>	
657	Invalid record_no.	
895	Record has already been accepted by Task Assignee	
1101	Empty or Invalid JSON data	If required input JSON is empty
1419	The Project Funding Sheet has to be created prior to this operation.	When user is creating a Fund type BP record or Funding and WBS type BP Record, but a Project Funding Sheet has not been created.
3000	Partial failure.	Partial failure of REST service
3002	Invalid Input	

Funding REST API Details

In This Section

Get Funds List 571
Create Fund 572
Update Fund 575
Delete Fund 576
Manual Fund Consumption 577
Get Fund Columns 583
Get Fund Assignment Order 585
Get Fund Assignment Order for Cost Sheet 586
Update Fund Assignment Order 588
Update Fund Assignment Order for Cost Sheet 589
Update Fund Status 590
Get Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio) 592
Update Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio) 597
Response Error Codes (Funding REST API Details) 603

Get Funds List

GET /ws/rest/service/v1/fund/{project_number}

Purpose:

Get all list of Funds

Input:

All parameters should be URL encoded.

Path Parameter

project_number: specify the project/shell number if you need to get project/shell level fund code. If no value is specified then company level fund code will be returned in the response.

url parameter -

filter=

{ "category":"All Bond Fund Sources",

"status":"Active"

}

Category filter is the display label of the category and is case sensitive.

One category can only be filter in a single request. Multiple categories filter is not supported.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

GetFunds


```
{
  "data":
  [
    {
        "id":3,
        "code":MaintainFund,
        "name":"MaintainFund",
        "category":"Grants",
        "description":"College Maintenance Fund",
        "status":"active"
      },
      {
        "id":4,
        "code":MiscFund,
        "fundname":"MiscFund",
        "fundcategory":"Grants",
        "description":"MiscellaenousFund",
        "status":"active"
      }
    ]
 ,
  "message":
  [
     "Success"
  ],
  "status":200
}
```


Create Fund

POST /ws/rest/service/v1/fund/{project_number}

Purpose:

Create Fund Codes.

If project number is specified and the fund doesn't exist at company level, throw Error.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the project/shell number to create the fund codes from company fund sheet. If no value is specified then the fund code will be created in company.

POST body is a JSON

Data JSON will be list of fund codes attributes.

> **Note**: POST call has input & output both as JSON in the body

For Company:

Create Fund - Company

```
{"data":[{
        "name":"MiscFund",
        "category":"Grants",
        "description":"Miscellaneous Fund",
         "upermitIssuerURL": "http://google.com test",
      }
]
}
```

Parentid is not accepted for tree mode. If provided, will be ignored.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Create Fund - Company

```
{
    "data": [
       {
         "id":4,
         "code":Fund6,
         "name":"Fund6",
```

```
        "category":"Grants",
        "description":"MiscellaenousFund",
        "upermitIssuerURL": "http://google.com test",
        "status":"active"
      }
    ],
    "message":
        [

          { code: "Fund5","status": "1206","message":"Fund Code is
referenced in lineitems"},
          { code: "Fund6","status": "200","message":"success"}
    ],
    "status": 200
}
```

For Projects, the list of fund codes are required to create fund codes.

Create Fund - Project

```
{"data":{
      "codes": ["Fund1","Fund2","Fund3"]
      }
}
```

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Create Fund - Project

```
{
    "data": {
        {
           "code": "Fund1",
           "id":31
        }
    },
    "message": [
```

```
          { code: "Fund2","status": "1201","message":"Fund Code used
already exists."},
          { code: "Fund1","status": "200","message":"success"}  ,
         { code: "Fund2","status": "1201","message":"Fund Code used
already exists."}
    ],
    "status": 3000
}
```

Multiple fund codes can be created in a single request. For Partial Success/Failure, status will be 3000.

Update Fund

PUT /ws/rest/service/v1/fund/

Purpose:

Update company fund code attributes.

Input:

All parameters should be URL encoded.

POST body is a JSON. Data will be list of fund code attributes to be updated at company level

> **Note**: POST call has input & output both as JSON in the body.

Update Fund

```
{"data":[{
        "code":"MiscFund",
        "category":"Grants",
        "description":"Miscellaneous Fund"
      }
]
}
```

Code , Name ,Id, Status and orderid cannot be updated. If provided, they will be ignored.

Output:

JSON object containing 'status', 'data', 'message'

A message will be present if the status is not 200 otherwise it will be "success".

Update Fund

```
{
    "data": [
          {
                "id":4,
                "code":MiscFund,
                "name":"MiscFund",
                "category":"Grants",
                "description":"MiscellaenousFund",
                "status":"active"
      }
    ],
    "message":
        [
          { code: "MiscFund","status": "200","message":"success"}
    ],
    "status": 200
}
```

Multiple fund codes can be updated in a single request.

Delete Fund

DELETE /ws/rest/service/v1/fund/{project_number}

Purpose:

Deletes Fund codes

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the project/shell number in which fund ids exists. If no value is specified then the fund ids will be deleted from company.

Path Parameter

Both input & output in JSON format in the body

Delete Fund

```
{
```

```
"data":{
"codes": ["Fund9","Fund5","Fund6"]
}
}
```

Output:

JSON object containing 'status', 'data', 'message'

Status codes are:

1> 200 OK , if all creation succeeds .

2> 3000 , If any of records creation fail.

For Partial failure, relevant message will be displayed to the fund code.

Delete Fund

```
{
    "data": {
    },
    "message": [
          { code: "Fund5","status": "1206","message":"Fund Code is
referenced in lineitems"},
          { code: ""Fund6"","status": "200","message":"success"}  ,
          { code: "Fund9","status": "1201","message":"Fund Code do not
exist"}
    ],
    "status": 3000
}
```

Manual Fund Consumption

POST /ws/rest/service/v1/fund/consumption/{project_number}

Purpose:

This API is used to perform the manual fund consumption on a record that is either in terminal state or non-terminal state. In case of transient state, this service can be used at both view only or action form of the BP.

Notes:

- Commitment funding is not supported by this service.
- Will support remove and replace of fund similar to UI.
- The Udesigner settings regarding funding option will also be validated as part of the service.

- All the BP records will be processed even if there are any failures in one of the record. Within the BP records all the line items will be processed. Failed record and line items will be reported in the response.
- The order of processing of the lineitem will be based on li_num value.
- This service will not rollup the amount to the fund sheet. Use update BP service or from UI when the BP is sent to next step or finish editing in case of Non workflow BP then the amount will be rolled up to the fund sheet.
- The integration user should have "Full Access" or "Update" permission under "Business Process Services" to run this service.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number of the funding sheet.

POST body contains the JSON object with array of BP assignments.

"data" is a JSON array which can have one or more BP records fund consumption data.

"bpname" (Required): Name of the BP corresponding to the BP record.

"record_no" (Required): The BP record number for which the fund consumption has to be processed.

"funds" is a JSON array which contains one or more fund consumption details.

"code" (Required) : Specify the fund code associated with project/shell or CBS.

"amount" (Required) : Datatype is Double. Specify the assignment value to be consumed from the fund.

"_bp_lineitems" : JSON Array to specify the BP lineitems. Required for "Project Level - Manual(Assign by BP Line Item)" and "CBS Level - Manual" assignment setting.

"li_num" (Required) : Use the "li_num" in "_bp_lineitems" from the response to "GET /ws/rest/service/v1/bp/record/" REST call with lineitem":"yes" as part of the request.

"uuu_tab_id" (Required) : Use the "uuu_tab_id" in"_bp_lineitems" from the response to "GET /ws/rest/service/v1/bp/record/" REST call with lineitem":"yes" as part of the request.

"group_id" (Required if lineitem has group else optional) : Use the "group_id" in"_bp_lineitems" from the response to "GET /ws/rest/service/v1/bp/record/" REST call with lineitem":"yes" as part of the request.

Sample Input Request

//Spends BP data is an example for Project Level - Manual(Assign by BP Record Total) assignment and //Payments BP data is an example for Manual (Assign by BP Line Item)" and "CBS Level - Manual" assignment.

```
{
   "data":[
      {
```

```
   "bpname":"Spends",
   "record_no":"uxsp-001",
   "funds":[
      {
         "code":"grants",
         "amount":300.25
      },
      {
         "code":"bonds",
         "amount":750.30
      }
   ]
},
{
   "bpname":"Payments",
   "record_no":"uxpa-001",
   "_bp_lineitems":[
      {
         "uuu_tab_id":"List of Contacts",
         "li_num":1,
         "group_id":60,
         "funds":[
            {
               "code":"grants",
               "amount":250.80
            },
            {
               "code":"bonds",
               "amount":750.40
            }
         ]
      },
      {
         "uuu_tab_id":"List of Contacts",
```

```
                "li_num":2,
                "group_id":60,
                "funds":[
                   {
                      "code":"grants",
                      "amount":250.40
                   },
                   {
                      "code":"bonds",
                      "amount":750.16
                   }
                ]
             },
             {
                "uuu_tab_id":"List of Contacts",
                "li_num":3,
                "group_id":60,
                "funds":[
                   {
                      "code":"fedral",
                      "amount":650.45
                   }
                ]
             }
          ]
       }
   ]
}
```


Output:

JSON object containing 'status', 'data', 'message'

"data" will be empty in the response.

"message" This will be an array of JSON object containing

"record_no" and "status". This will "success" or error message for the failure.

"status" contains:

- 200 when all the record's fund consumption completed successfully and
- 3000 if any one record fund consumption has failed.

Sample Output Response


```
{
   "data":[
      {
         "record_no":"uxsp-001",
         "fund_consumed":500.0
      },
      {
         "record_no":"uxpa-001",
         "_bp_lineitems":[
            {
               "uuu_tab_id":"List of Contacts",
               "li_num":1,
               "cbs_code":"1_D_D",
               "fund_consumed":500.0
            },
            {
               "uuu_tab_id":"List of Contacts",
               "li_num":2,
               "cbs_code":"2_D_D",
               "fund_consumed":450.0
            }
         ]
      }
   ],
   "message":[
      {
         "record_no":"uxsp-001",
         "funds":[
            {
               "code":"grants",
```

```
            "status":1214,
            "message":"Fund assignment amount is required."
         },
         {
            "code":"bonds",
            "status":200,
            "message":"Success"
         }
      ]
   },
   {
      "record_no":"uxpa-001",
      "_bp_lineitems":[
         {
            "uuu_tab_id":"List of Contacts",
            "li_num":1,
            "funds":[
               {
                  "code":"grants",
                  "status":1214,
                  "message":"Fund assignment amount is required."
               },
               {
                  "code":"bonds",
                  "status":200,
                  "message":"Success"
               }
            ]
         },
         {
            "uuu_tab_id":"List of Contacts",
            "li_num":2,
            "funds":[
               {
```

```
                    "code":"grants",
                    "status":200,
                    "message":"Success"
                },
                {
                    "code":"bonds",
                    "status":200,
                    "message":"Success"
                }
            ]
        }
    ]
  }
],
"status":3000
}
```


Get Fund Columns

GET /ws/rest/service/v1/fund/column/{project_number}

Purpose:

Get the funding column data. This API can be used to get the fund balance as user can use a formula column to calculate the balance.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Optional): Specify the project/shell number to get the project/shell funding data or empty to get the company data.

url parameter -

filter=

{

"column_names":["Manual Funding by Project" ,"Project Funding"],

"codes":["Fund0","Fund1","Fund2"]

}

Notes:

- Filter can be used to filter on column name and the fund code to get the data.
- For funding sheets with tree as display mode, if parent fund code is given in filter condition then error will be thrown.
- If with given filter column name, multiple columns exists in the funding sheet then validation error will be thrown.

Output:

JSON object containing 'status', 'data', 'message'

> **Note**: If there are no columns defined in the fund sheet then the service will return an empty data array in the response.

Sample Response


```
{
  "data":[
     {
        "Project Funding":{
        "Data Source" : "Project_Funding",
        "Funds":{
                  "fund0":101.00,
                  "fund1":100.2,
                  "fund2":100.3
                }
        }
     },
     {
        "Manual Funding by Project":{
        "Data Source" : "Manual_Funding",
        "Funds":{
                  "fund0":101.00,
                  "fund1":100.2,
                  "fund2":100.3
                }
        }
     }
  ],
  "message":[
```

```
    "success"
  ],
  "status":"200"
}
```

Partial get is not allowed i.e. if one of the filter conditions fails then only error message will be returned.

Get Fund Assignment Order

GET /ws/rest/service/v1/fund/order/{project_number}

Purpose:

Get Fund Order

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number of the funding sheet.

Output:

JSON object containing 'status', 'data', 'message'

Status	Message	Condition
200	success	If all the data is retrieved in the response
500	exception message	If there is any exception when running this service
1240	Request is valid for Shell or Project, not for Program.	If the GET request is called for Program
1207	Funding sheet do not exist.	If shell does not have funding sheet

Get Fund Order

```
{
   "data": [{
     "code_order": ["f1001", "f1002", "f1003"]
      }],
"message": [
        "success"
    ],
```

```
    "status": 200
}
```

Get Fund Assignment Order for Cost Sheet

GET /ws/rest/service/v1/cost/fund/order/{project_number}

Purpose:

Get Fund Order

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number of the funding sheet

optional url parameter -

```
filter=
{
"cbs_codes":["code1" ,"code2"]
}
```

If cbs_codes in filter do not have fund assigned to them , then empty code_order List will return in response for those cbs_codes.

When filter is not provided then only those cbs_codes which have fund assigned to them , will return in response.

Output:

JSON object containing 'status', 'data', 'message'

Status	Message	Condition
200	success	If all the data is retrieved in the response
500	exception message	If there is any exception when running this service
1240	Request is valid for Shell or Project, not for Program.	If the GET request is called for Program
1207	Funding sheet do not exist.	If shell does not have funding Sheet
707	Cost sheet does not exist.	If shell does not have Cost Sheet

Get Fund Order

Without filter condition: response contains only those cbs_codes which have fund assigned.

```
{
```

```
   "data": [{
            "cbs_code": "cost_code_1",
            "code_order": ["f1001", "f1002", "f1003"]
        },
        {
            "cbs_code": "cost_code_2",
            "code_order": ["f1002", "f1004", "f1003"]
        }
 ],
"message": [
        "success"
    ],
    "status": 200
}
```


With filter condition: returns empty code_order List for "cost_code_3". It does not have fund assigned.


```
{
   "data": [{
            "cbs_code": "cost_code_1",
            "code_order": ["f1001", "f1002", "f1003"]
        },
        {
            "cbs_code": "cost_code_2",
            "code_order": ["f1002", "f1004", "f1003"]
        },
        {
            "cbs_code": "cost_code_3",
            "code_order": []
        }
 ],
"message": [
        "success"
    ],
    "status": 200
```

```
}
```

Update Fund Assignment Order

PUT /ws/rest/service/v1/fund/order/{project_number}

Purpose:

Fund ordering to be used for fund assignment when auto order rule is specified for the business process. this API will allow user to change the fund order in the project/shell funding sheet.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number of the funding sheet.

POST body (JSON)

The data object will contain all the fund codes listed in order.

Order Fund Codes

Sample Input

```
{
    "data": [{
        "code_order": ["1001", "1002", "1003"]
    }]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Status	Message	Condition
200	success	If all the data is retrieved in the response

Sample Output

```
{
    "data": [{
        "code_order": ["1001", "1002", "1003"]
    }]
,
"message": [
        "success"
    ],
```

```
    "status": 200
}
```

Output:

JSON object containing 'status', 'data', 'message'

If update for one code fails, all the changes will be rolled back.

Update Fund Assignment Order for Cost Sheet

This PUT /ws/rest/service/v1/cost/fund/order/{project_number}

Purpose:

Fund ordering to be used for fund assignment when auto order rule is specified for the business process. this API will allow user to change the fund order in the project/shell funding sheet.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Required): Specify the project/shell number of the funding sheet.

POST body (JSON)

The data object will contain the list of all fund codes for a given cbs code listed in order.

Order Fund Codes by CBS codes

Sample Input

```
{
   "data": [
        {
            "cbs_code": "00000~~00500",
            "code_order": ["f1001", "f1002", "f1003"]
        },
        {
            "cbs_code": "00000~~005001",
            "code_order": ["f1002", "f1004", "f1003"]
        }
   ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Sample Output

```
{
    "data": [{
            "cbs_code": "00000~~00500",
            "code_order": ["f1001", "f1002", "f1003"]
        }
    ],
    "message": [  {
            "code": "00000~~005001",
            "message": "CBS Code is invalid.",
            "status": 1241
        },
        {
            "code": "00000~~00500",
            "message": "success",
            "status": 200
        }
    ],
    "status": 3000
}
```

Partial update is allowed.

Update Fund Status

PUT /ws/rest/service/v1/fund/status/{project_number}

Purpose:

API will allow user to change the fund status in the project/shell funding sheet.

Input:

All parameters should be URL encoded.

Path Parameter

project_number(Optional): Specify the project/shell number of the funding sheet. If not specified,will update company fund codes.

POST body (JSON)

The data object will have fund ids and the status to be updated to. It will take the id as the key and the status as value.

Sample input request

```
{
   "data": [
        {"code":"1001","status":"Active"},
        {"code":"1002","status":"Open"},
        {"code":"1003","status":"Inactive"}
   ]
}
```

Output:

JSON object containing 'status', 'data', 'message'

Update Status

```
{
    "data": [
        {
            "code": "1003",
            "status": "Inactive"
        }
    ],
    "message": [
        {
            "code": "1001",
            "message": "Fund Code do not exist.",
            "status": "1203"
        },
        {
            "code": "1002",
            "message": "Status field value is not correct.",
            "status": "663"
        },
        {
            "code": "1003",
            "message": "success",
            "status": "200"
        }
```

```
    ],
    "status": 3000
}
```

Partial update is allowed.

Get Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio)

GET /ws/rest/service/v1/fund/sov/consumption/{project_number}

Purpose:

To Fetch assigned Funds to CBS in SOV of types "General Spends" & "Payment Applications".

Input Parameter:

All parameters should be URL encoded.

Both input & output in JSON format.

Get Fund Consumption for CBS Level SOV input JSON

```
filter = {
    "commit_bp_name" : ["bp_name"],
    "record_no" : ["base_record_no"],
    "cbs_code" : ["100~~101"],
    "item#" : [1234],
    "sov_type" : "General Spends",
    "assignment_type": "SOV Auto-Order"
}
```

If a filter is not provided, then all SOVs (from "General Spends" & "Payment Applications") will be fetched. If a filter is provided, then the "AND" condition is seen.

Filter condition key definitions

Key	Definition
commit_bp_name	Optional. The commit_bp_name references those BPs that have SOV Auto-Order, or SOV Auto-Ratio, fund assignment selected in Funding Sheet properties. One or more inputs are supported in filter .

record_no	Optional. The base record number of the SOV sheet. One or more inputs are supported in filter . The record_no supports '%' wildcard character to filter record_no. For example: The "record_no" : "PO-001%", will fetch SOV details for all of the records starting with "PO-001".
cbs_code	Optional. The CBS code to which the fund is assigned. One or more inputs are supported in filter.
item#	Optional. Uniquely identifiable row number "item#" in the SOV sheet. One or more inputs are supported in filter.
sov_type	Optional. Only one value is supported at a time in the filter: General Spends" or "Payment Applications".
assignment_type	Optional. Only one value is supported at a time in the filter: "SOV Auto-Order" or "SOV Auto-Ratio".

Output:

JSON object containing 'status', 'data', 'message'

Get Fund Consumption for CBS Level SOV output JSON


```
{
  "data": {
    "Purchase Orders": {
      "sov_type": "General Spends",
      "assignment_type": "SOV Auto-Ratio",
      "record_no": {
        "PO-0002": {
          "rows": [
```

```
{
  "item#": "000010",
  "cbs_code": "100~~101",
  "fund_assignment": [
    {
      "fund_code": "funding",
      "fund_ratio": "40",
      "status" : "Active"
    },
    {
      "fund_code": "State Funding",
      "fund_ratio": "0",
      "status" : "Inactive"
    },
    {
      "fund_code": "Federal Funding",
      "fund_ratio": "60",
      "status" : "Active"
    }
  ]
},
{
  "item#": "000020",
  "cbs_code": "100~~102",
  "fund_assignment": [
    {
      "fund_code": "State Funding",
      "fund_ratio": "100",
      "status" : "Active"
    }
  ]
},
{
  "item#": "000030",
```

```
          "cbs_code": "100~~102",
          "fund_assignment": []
        },
        {
          "item#": "000040",
          "cbs_code": "300~~302",
          "fund_assignment": []
        }
      ]
    },
    "PO-0001": {
      "rows": [
        {
          "item#": "000010",
          "cbs_code": "100~~101",
          "fund_assignment": [
            {
              "fund_code": "funding",
              "fund_ratio": "100",
              "status" : "Active"
            }
          ]
        }
      ]
    }
  }
},
,
"Group_by_CBS_base_commit_bp": {
  "sov_type": "General Spends",
  "assignment_type": "SOV Auto-Order",
  "record_no": {
    "uxfund20-0000": {
      "rows": [
```

```
          {
            "item#": "000010",
            "cbs_code": "100~~101",
            "fund_assignment": [
              {
                  "fund_code" : "funding",
                  "status" : "Active"
              },
              {
                  "fund_code" : "State Funding",
                  "status" : "Active"
              }
              {
                  "fund_code" : "Federal Funding",
                  "status" : "Active"
              }
            ]
          }
        ]
      }
    }
  },
  "SM Cost PG Commit WF": {
    "sov_type": "Payment Applications",
    "assignment_type": "SOV Auto-Order",
    "record_no": {
      "uxfund4-0001": {
        "rows": [
          {
            "item#": "000010",
            "cbs_code": "100~~101",
            "fund_assignment": [
                  {
                      "fund_code" : "funding",
```

```
                    "status" : "Active"
                }
            ]
          }
        ]
      }
    }
  }
  },
  "message": [
    "Success"
  ],
  "status": 200
}
```


Notes:

- To perform this call, the Integration user need to have the "Cost CBS Services > Get" permission (under "User Administration in Company admin mode").
- Only the SOV sheets with "Fund Assignment" enabled (in the sheets) will be fetched as part of GET SOV consumption call.
- The SOV sheets that are funded from Commit Fund Sheet will not get fetched from this GET call.
- From the SOV sheet, the Summary rows will not be fetched. Only the rows with Cost code assigned to them (normal rows or breakdown of summary row) will be fetched in GET call.

Status codes are:

1>200, for success

Update Fund Consumption for CBS Level SOV (Auto-Order / Auto-Ratio)

PUT /ws/rest/service/v1/fund/sov/consumption/{project_number}

Purpose:

To assign Funds to CBS in SOV of types "General Spends" & "Payment Applications".

Input:

All parameters should be URL encoded.

Both input & output in JSON format in the body.

Fund Consumption for CBS Level SOV input JSON

SOV Auto-Ratio

```
{
   "data":{
      "Purchase Orders" ( commit_bp_name ):{
         "sov_type":"General Spends", ( optional )
         "assignment_type":"SOV Auto-Ratio", ( optional )
         "record_no":{
            "PO-0002" ( SOV base record number ):{
               "rows":[
                  {
                     "item#":"000010",
                     "cbs_code":"100~~101", ( optional )
                     "fund_assignment":[
                        {
                           "fund_code":"funding",
                           "fund_ratio":"40",
                           "status" : "Active" ( optional )
                        },
                        {
                           "fund_code":"State Funding",
                           "fund_ratio":"0"
                           "status" : "Active"
                        },
                        {
                           "fund_code":"Federal Funding",
                           "fund_ratio":"60"
                        }
                     ]
                  },
                  {
                     "item#":"000020.00001",
                     "breakdown" : "BD1",
                     "cbs_code":"100~~102",
                     "fund_assignment":[
                        {
```

```
                        "fund_code":"State Funding",
                        "fund_ratio":"100"
                     }
                  ]
               }
            ]
         },
         "PO-0001":{
            "rows":[
               {
                  "item#":"000010",
                  "cbs_code":"100~~101",
                  "fund_assignment":[
                     {
                        "fund_code":"funding",
                        "fund_ratio":"100"
                     }
                  ]
               }
            ]
         }
      }
   }
}
```

SOV Auto-Order (fund_assignment can be provided in two ways):

1)

```
   {
       "data":{
         "Group_by_CBS_base_commit_bp" ( commit_bp_name ): {
           "sov_type": "General Spends",
           "assignment_type": "SOV Auto-Order",
           "record_no": {
               "uxfund20-0000" ( SOV base record number ): {
               "rows": [
```

```
                    {
                    "item#": "000010",
                    "cbs_code": "100~~101",
                    "fund_assignment": [
                        "funding",
                        "State Funding",
                        "Federal Funding"
                    ]
                    }
                ]
                }
            }
          }
       }
   }
```

2)

```
   {
       "data":{
           "Group_by_CBS_base_commit_bp": {
               "sov_type": "General Spends",
               "assignment_type": "SOV Auto-Order",
               "record_no": {
                   "uxfund20-0000": {
                       "rows": [
                           {
                               "item#": "000010",
                               "cbs_code": "100~~101",
                               "fund_assignment": [
                                   {
                                       "fund_code" : "funding"
                                   },
                                   {
                                       "fund_code" : "State Funding"
                                   },
                                   {
                                       "fund_code" : "Federal Funding"
                                   }
                               ]
                           }
                       ]
```

```
                    }
                }
            }
        }
    }
```

Notes:

The "item#" is mandatory, to uniquely identify row to be updated, in SOV sheet.

For SOV Auto-Order , if only one fund_code is provided in input without fund_ratio, then default 100% will be applied to that fund_code.

SOV sheets configured to be funded from Commit Fund Sheet, will not get updated from this update call.

- To perform this call, the Integration user need to have the "Cost CBS Services > Get" permission (under "User Administration in Company admin mode").
- The update operation supports only one Commitment Business process in Input.
- Only the SOV sheets with "Fund Assignment" enabled (in the sheets) will be updated.
- In the SOV sheet, only the rows that have Cost code assigned to them are provided in input JASON.
- If a row is divided in breakdowns, then both the "item#" and the "breakdown" values must be provided for identification in order for the row to be updated.For the SOV Auto-Order, if only one fund_code is provided in input, without fund_ration, then the default value of 100% will be applied to that fund_code.
- The SOV sheets that have been configured to be funded from the Commit Fund Sheet will not get updated from this update call.

Output:

JSON object containing 'status', 'data', 'message'

Response message contains key as "<record_no>/<item#>/<breakdown, if present>

And "item_status", "item_message" contains individual response for that row in SOV record.

Fund Consumption for CBS Level SOV ouput JSON

SOV Auto-Ratio

```
{
   "data":[
   ],
   "message":[
      {
         "PO-0002/000010" :
        {
                "item_status":200,
                "item_message":"success"
```

```
        }
        "PO-0002/000020.00001/BD1" :
        {
                "item_status":200,
                "item_message":"success"
        },
        "PO-0001/000010" :
        {
                "item_status":200,
                "item_message":"success"
         }
      }
   ],
   "status":200
}
```

SOV Auto-Order

```
{
   "data":[
   ],
   "message":[
      {
         "uxfund20-0000/000010" :
        {
                  "item_status":200,
                  "item_message":"success"
        }
      }
   ],
   "status":200
}
```

Status codes are:

1>200, for success

2> 3000, for Partial success.

Response Error Codes (Funding REST API Details)

Code	Message	Details
200	success	If the request is successful
500	Server Error, contact the system administrator.	If there is any exception when running this service
505	token+" is not present, it is required in the formula:"+value+" for field "+name	If the data provided is not correct like pull down with a label which does not exist.
602	Project/Shell Number is not correct	project number is incorrect or does not exists
603	Business Process Name is not correct	"bpname" is not specified in the input JSON and or is invalid value
651	CBS Code is not present.	"cbs_code" is not present in the input JSON
657	Invalid record_no.	"record_no" value is not specified in the input JSON or is invalid value.
663	Status field value is not correct.	If status value is not valid i.e it is neither active nor inactive
701	Invalid Column Name	If column name given in the filter condition does not belong to funding sheet
707	Cost sheet does not exist.	If shell does not have Cost Sheet
811	Invalid Filter Condition.	
859	This record is already terminated.	When BP specified in the input JSON is terminated.
1101	Empty or Invalid JSON data.	Input request JSON is invalid
1202	Fund Code cannot be deleted as child funds exist.	
1203	Fund Code do not exist.	If the input fund code does not belong to project funding sheet or configured in cost sheet for CBS
1204	Fund Code is referenced in lineitems.	Fund Code is referenced in lineitems and has data.

1205	Fund Code has funds allocated manually.	
1206	This operation cannot be performed on Root/Unassigned/Parent Fund Code	
1207	Funding sheet do not exist	project fund sheet is not defined
1208	Fund Code is not valid.	
1209	Fund Code is required	When input JSON didn't contain "code" parameter in "funds" array.
1210	Only manual funding assignment rule is supported	if the assignment rule for BP is not set for manual funding or it is a commitment funding BP.
1211	Only General Spends, Payment Application or Generic Cost BP are supported	BP name specified in the input is not a spends, payment or generic cost BP type
1212	code not present in input request	
1213		If Fund status is inactive
1214	Fund assignment amount is required.	
1215	Business Process line item is required.	"_bp_lineitems" is not specified in the input JSON for lineitem or CBS assignment
1216	li_num is not present	"li_num" is not present in the input JSON
1217	li_num is Invalid.	"li_num" has invalid value like not a number or it doesn't belong to the BP lineitem
1218	Invalid CBS Code for lineitem.	For CBS manual assignment rule the "cbs_code" and "li_num" values in input JSON does not match
1219	Project status is Inactive.	
1220	Fund code cannot be empty	
1221	Consumption cannot be done on a negative fund balance	When the fund balance is negative
12	Amount cannot be greater than fund	When input amount is greater than

22	balance	available fund balance
12 23	Amount must be between	validation message based on the input amount and total fund amount and other values
12 24	Amount cannot be greater than unassigned amount	Input amount is greater than unassigned amount
12 25	Total amount must be between	Input amount is less or greater than the lineitem or record amount
12 26	Funds are not provided for consumption.	"funds" array is not specified in the input JSON or it is empty
12 27	Fund consumption is not allowed.	If funding option is disabled in uDesigner settings.
12 28	Fund Code is used in Projects.	
12 29	Fund amount is required.	"amount" value is not specified in the input JSON
12 30	Fund amount is invalid.	"amount" value is invlid in input JSON
12 31	Separator cannot be included in the value.	
12 32	Fund consumption failed.	When there is an exception or error when processing the fund consumption
12 33	No Fund Codes are assigned to CBS for this lineitem	If the CBS code provided in the input JSON does not have any fund code associated with it.
12 34	Fund Code is not assigned to this CBS	If Fund Code is not assigned to CBS
12 35	Multiple Fund Columns with same name exists	If multiple columns of a fund sheet exists with same column name.
12 36	Tab name is invalid.	tab name specified in uuu_tab_id parameter is not a valid value.
12 37	Fund codes specified in the given project are not valid:	Fund codes specified are in valid.
12 38	All the fund codes in the project are not specified.	All the fund codes in the project are not specified.
12 39	Fund Code specified is duplicate:	Fund code specified is duplicate
14 09	Invalid fund_assignment.	

14 10	Invalid funds: <fund names >	
14 11	Inactive funds: <fund names>	
14 12	Invalid fund_ratio : <input value>	
14 13	Sum of fund_ratio is not 100% : < total value >	
14 14	Invalid item# or breakdown.	
14 15	Fund Assignment not enabled.	
14 16	Invalid Commitment Business process.	
12 40	Request is valid for Shell or Project, not for Program.	Request is valid for Shell or Project, not for Program.
12 41	CBS Code is invalid.	CBS Code is invalid.
12 42	All the fund codes in the cbs code are not specified.	All the fund codes assigned in the cbs code are not specified.
12 43	Fund consumption cannot be performed when the project/shell status is Inactive or View-Only or On-Hold.	Fund consumption is allowed only active project/shell status.
12 44	Invalid Fund Category, Please provide correct Category.	Invalid Fund Category, Please provide correct Category.
12 45	Update cannot be performed when project/shell status is Inactive or View-Only or On-Hold.	
12 46	group_id is Invalid.	group_id value specified in fund consumption call for lineitems is invalid.
12 47	group_id is required.	group_id value is required for lineitem BP with groups when invoking the fund consumption call.
30 00	Partial failure.	Partial failure of REST service
30 02	Invalid input	If input JSON didn't contain the "data" parameter
30 02	Only one business process is supported.	

3002	No data in input JSON.	
3003	Business Process is not Active.	When fund consumption is performed on a inactive BP.

Non-Workflow BP Permissions

> **Note**: Ensure that the integration user has the required Business Process Permissions.

In This Section

Get BP Records Permission 609
Update/Modify BP Records Permission 611
Create (Add User/Group) BP Records Permission 616
Delete (Remove User/Group) BP Records Permission 619

Get BP Records Permission

POST /ws/rest/service/v1/bp/record/permission/list/{project_number}

Purpose:

Get non-workflow records permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.

For one or more records

```
{
   "bpname" : "Vendors",
   "record_no" : ["VEN-0023","VEN-0024"],
    "filter_condition" : "status=Active"
}
```

For only one record

```
{
```

```
   "bpname" : "Vendors",
   "record_no" : "VEN-0023",
    "filter_condition" : "status=Inactive"
}
```

"bpname" is mandatory input parameter.

In the request body record_no can be provided in below two formats:

1. "record_no": ["VEN-0023","VEN-0024"] For one or more records.

2. "record_no": "VEN-0023" For only one record.

"record_no" is optional. If no value is specified then all the BP record permission will be returned.

"filter_condition" is optional parameter and filter is supported only on "status" field. It will be "AND" condition between "record_no" & "filter_condition".

Output:

JSON object containing 'status', 'data', 'message'

Message will be present if status is not 200 otherwise it will be "success".

Get BP Permission Sample Response


```
{
    "data": {
        "VEN-0024": {
            "record_status": "Active",
            "permissions": [
                {
                    "login_name": "coadmin",
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "U"
                },
                {
                    "full_name": "Space Planners",
                    "group_name": "Space Planners",
```

```
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "CG"
                }
            ]
        },
        "VEN-0023": {
            "record_status": "Terminated",
            "permissions": [
                {
                    "login_name": "coadmin",
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                }
            ]
        }
    },
    "message": [
        "success"
    ],
    "status": 200
}
```


Update/Modify BP Records Permission

PUT /ws/rest/service/v1/bp/record/permission/{project_number}

Purpose:

Update non-workflow records permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.

```
{
    "options":{
        "bpname" : "simple1"
    },
    "data": {
        "record_no": ["uxss1-0007", "uxss1-0006", "uxss1-0005"]
        "permissions": [
                {
                    "login_name": "coadmin",
                    "type": "U",
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "modify_permission": "0",
                        "view": "1"
                    }
                }
                {
                    "full_name": "Building Managers",
                    "group_name": "Building Managers",
                    "permission": {
                        "edit_data": "1",
```

```
                    "view": "1",
                    "modify_permission": "0"
                },
                "type": "PG"
            }
        ]
    }
}
```

Here "bpname" & "record_no" are mandatory in options.

In the request body record_no can be provided in below two formats

1. "record_no": ["uxss1-0007", "uxss1-0006", "uxss1-0005"] For one or more records.

2. "record_no": "uxss1-0007" For only one record.

If record_no not provided, then error response will be provided to user to provide record_number.

Output:

JSON object containing 'status', 'data', 'message'

Message will be present if status is not 200 otherwise it will be "success".

Update/Modify BP Records Permission Sample Response

```
{
    "data": {
        "uxss1-0007": {
            "record_status": "Active",
            "permissions": [
                {
                    "login_name": "firstname",
                    "full_name": "Firstname Lastname",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                },
```

```
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "0"
                },
                "type": "U"
            },
            {
                "full_name": "Building Managers",
                "group_name": "Building Managers",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "0"
                },
                "type": "PG"
            }
        ],
        "updated_users": [
            "coadmin"
        ],
        "updated_groups": [
            "Building Managers"
        ]
    },
    "uxss1-0005": {
        "record_status": "Terminated",
        "permissions": [
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
```

```
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "U"
            }
        ],
        "updated_users": [],
        "updated_groups": []
    },
    "uxss1-0006": {
        "record_status": "Active",
        "permissions": [
            {
                "login_name": "coadmin",
                "full_name": "Company Administrator",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "0"
                },
                "type": "U"
            },
            {
                "login_name": "firstname",
                "full_name": "Firstname Lastname",
                "permission": {
                    "edit_data": "1",
                    "view": "1",
                    "modify_permission": "1"
                },
                "type": "U"
            },
```

```
                {
                    "full_name": "Building Managers",
                    "group_name": "Building Managers",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "0"
                    },
                    "type": "PG"
                }
            ],
            "updated_users": [
                "coadmin"
            ],
            "updated_groups": [
                "Building Managers"
            ]
        }
    },
    "message": [
        "success"
    ],
    "status": 200
}
```


Create (Add User/Group) BP Records Permission

POST /ws/rest/service/v1/bp/record/permission/{project_number}

Purpose:

Add User/Group to non-workflow records permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.

```
{
    "options":{
        "bpname" : "Vendors"
    },
    "data": {
        "record_no": ["VEN-002598"],
        "permissions": [
                {
                    "login_name": "PU1",
                    "type": "U",
                    "full_name": "P1 user",
                    "permission": {
                        "edit_data": "1",
                        "modify_permission": "1",
                        "view": "1"
                    }
                }
            ]
        }
}
```

Here "bpname" & "record_no" are mandatory in options.

In the request body record_no can be provided in below two formats.

1. "record_no": ["uxss1-0007", "uxss1-0006", "uxss1-0005"] For one or more records.

2. "record_no": "uxss1-0007" For only one record.

If record_no not provided, then error response will be provided to user to provide record_number.

Output:

JSON object containing 'status', 'data', 'message'

Message will be present if status is not 200 otherwise it will be "success".

Create (Add User/Group) BP Records Permission Sample Response


```
{
    "data": {
        "VEN-0025": {
            "record_status": "Active",
            "added_users": [PU1],
            "permissions": [
                {
                    "login_name": "PU1",
                    "full_name": "PU1 P",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                },
                {
                    "login_name": "coadmin",
                    "full_name": "Company Administrator",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
                        "modify_permission": "1"
                    },
                    "type": "U"
                },
                {
                    "full_name": "Space Planners",
                    "group_name": "Space Planners",
                    "permission": {
                        "edit_data": "1",
                        "view": "1",
```

```
                    "modify_permission": "1"
                },
                "type": "CG"
            }
        ],
        "added_groups": []
    }
},
"message": [
    "success"
],
"status": 200
}
```


Delete (Remove User/Group) BP Records Permission

DELETE /ws/rest/service/v1/bp/record/permission/{project_number}

Purpose:

Add User/Group to non-workflow records permission (in user mode) of a specific BP in a shell based on shell number or from Company level if project/shell number is not provided.

The input JSON shall provide various options to be considered for fetching the data.

Input:

All parameters should be URL encoded.

Path Parameter

project_number: Specify the Project number in which the records exists, if not provided then records are considered to be fetched from Company Level.

POST body is a JSON

> **Note**: POST call has input & output both as JSON in the body.


```
{
    "options":{
        "bpname" : "Vendors"
    },
    "data": {
        "record_no": ["VEN-002589"],
```

```
        "user_names" : ["coadmin","donna","PU1"]
      }
}
```

Here "bpname" & "record_no" are mandatory in options.

In the request body record_no can be provided in below two formats

1. "record_no": ["uxss1-0007", "uxss1-0006", "uxss1-0005"] For one or more records.

2 "record_no": "uxss1-0007" For only one record.

If record_no not provided, then error response will be provided to user to provide record_number.

Output:

JSON object containing 'status', 'data', 'message'

Message will be present if status is not 200 otherwise it will be "success".

Delete (Remove User/Group) BP Records Permission Sample Response

```
{
    "data": {
        "VEN-0025": {
            "deleted_groups": [],
            "record_status": "Active",
            "deleted_users": [
                "PU1 P"
            ]
        }
    },
    "message": [
        "success"
    ],
    "status": 200
}
```

Currency

When importing Business Processes (BPs), you often must specify a project currency that Unifier can use for cost BPs and other calculations. You may need to enter the code corresponding to the currency used where the project is located. The currency can also be set in a user preference template.

The following tables list the supported currencies and their supporting codes.

In This Section

Currencies and codes (A-G) 621
Currencies and codes (H-N) 623
Currencies and codes (O-T) 625
Currencies and codes (U-Z) 626

Currencies and codes (A-G)

Currency Name	Code
Afghani	AFN
Algeria Dinar	DZD
Ariary	MGA
Australia Dollar	AUD
Azerbaijanian Manat	AZN
Bahamas Dollar	BSD
Bahrain Dinar	BHD
Barbados Dollar	BBD
BCEAO Franc	XOF
Belarus Ruble	BYR
Belize Dollar	BZD
Bermuda Dollar	BMD
Bhutan Ngultrum	BTN
Bolivia Boliviano	BOB
Botswana Pula	BWP
Brazil Real	BRL
Burundi Franc	BIF

Currency Name	Code
Canada Dollar	CAD
Cape Verde Escudo	CVE
Cayman Isl ds Dollar	KYD
CFA Franc BEAC	XAF
Chile Peso	CLP
Chile UF	CLF
China Yuan Renminbi	CNY
Colombia Peso	COP
Comoros Franc	KMF
Comptoirs Francais du Pacifique Franc	XPF
Congo-Kinshasa Franc	CDF
Convertible Marka	BAM
Costa Rica Colon	CRC
Croatia Kuna	HRK
Cuba Peso	CUP
Cyprus Pound	CYP
Czech Republic Koruna	CZK
Denmark Krone	DKK
Dirham	AED
Djibouti Franc	DJF
Dobra	STD
Dominican Republic Peso	DOP
Dram	AMD
East Caribbean Dollar	XCD
Egypt Pound	EGP
El Salvador Colon	SVC
Eritrea Nakfa	ERN
Estonia Kroon	EEK
Ethiopia Birr	ETB
Euro	EUR
Falkland Islands Pound	FKP

Currency Name	Code
Fiji Dollar	FJD
Gambia Dalasi	GMD
Georgia Lari	GEL
Ghana New Cedi	GHs
Gibraltar Pound	GIP
Guatemala Quetzal	GTQ
Guilder	AWG
Guinea Franc	GNF
Guyana Dollar	GYD

Currencies and codes (H-N)

Currency Name	Code
Haiti Gourde	HTG
Honduras Lempira	HNL
Hong Kong Dollar	HKD
Hungary Forint	HUF
Iceland Krona	ISK
Indian Rupee	INR
Indonesia Rupiah	IDR
Iran Rial	IRR
Iraq Dinar	IQD
Israel New Shekel	ILS
Jamaica Dollar	JMD
Jordan Dinar	JOD
Kazakhstan Tenge	KZT
Kenya Shilling	KES
Kuwait Dinar	KWD
Kwanza	AOA
Laos Kip	LAK

Currency Name	Code
Latvia Lat	LVL
Lebanon Pound	LBP
Lek	ALL
Lesotho Loti	LSL
Lev	BGN
Liberia Dollar	LRD
Libya Dinar	LYD
Lithuania Litas	LTL
Macau Pataca	MOP
Macedonia Denar	MKD
Malawi Kwacha	MWK
Malaysia Ringgit	MYR
Maldives Rufiyaa	MVR
Malta Lira	MTL
Mauritania Ouguiya	MRO
Mauritius Rupee	MUR
Mexico Peso	MXN
Mongolia Tughrik	MNT
Morocco Dirham	MAD
Myanmar Kyat	MMK
Namibia Dollar	NAD
Nepal Rupee	NPR
Netherlands Antilles Guilder	ANG
New Mozambique Metical	MZN
New Romania Leu	RON
New Turkish Lira	TRY
New Zealand Dollar	NZD
Nicaragua Gold Cordoba	NIO
Nigeria Naira	NGN
North Korea Won	KPW
Norway Krone	NOK

Currencies and codes (O-T)

Currency Name	Code
OmanRial	OMR
Paanga	TOP
Pakistan Rupee	PKR
PanamaBalboa	PAB
Papua New Guinea Kina	PGK
Paraguay Guarani	PYG
Peru Nuevo Sol	PEN
Peso	ARS
Philippines Peso	PHP
Poland Zloty	PLN
Pound Sterling	GBP
Qatar Riyal	QAR
Riel	KHR
Russia Ruble	RUB
Rwanda Franc	RWF
Saint Helena Pound	SHP
Saudi Arabia Riyal	SAR
Serbian Dinar	RSD
Seychelles Rupee	SCR
Sierra Leone Leone	SLL
Singapore Dollar	SGD
Slovakia Koruna	SKK
Slovenia Tolar	SIT
Solomon Islands Dollar	SBD
Som	KGS
Somalia Shilling	SOS
South Africa Rand	ZAR
South Korea Won	KRW

Currency Name	Code
South Sudanese Pound	SSP
Sri Lanka Rupee	LKR
Sudanese Dinar	SDG
Suriname Dollar	SDG
Sweden Krona	SEK
Switzerland Franc	CHF
Syria Pound	SYP
Taiwan New Dollar	TWD
Taka	BDT
Tala	WST
Tanzania Shilling	TZS
Thailand Baht	THB
Transnistria Moldova Leu	MDL
Trinidad and Tobago Dollar	TTD
Tunisia Dinar	TND
Turkmenistan Manat	TMM
Tuvalu Dollar	TVD

Currencies and codes (U-Z)

Currency Name	Code
Uganda Shilling	UGX
Ukraine Hryvna	UAH
United States Dollar	USD
Uruguay Peso	UYU
Uzbekistan Som	UZS
Vatu	VUV
Venezuela Bolivar Fuerte	VEF
Vietnam Dong	VND
Yemen Rial	YER

Currency Name	Code
Yen	JPY
Zambia Kwacha	ZMK
Zimbabwe Dollar	ZWD

Time Zone

When importing user records, you must enter a starting time zone that Unifier can use for time stamps. You may wish to enter the code corresponding to the company office where the user is located. Time zone is a user preference setting that the user can change later if necessary. The time zone can also be set in a user preference template.

About UTC

The Coordinated Universal Time (UTC) is a 24-hour time standard. The westernmost time zone uses UTC−12, being twelve hours behind UTC; the easternmost time zone, theoretically, uses UTC+12, being twelve hours ahead of UTC. Calculation of the UTC time standard is the same as with Greenwich Mean Time (GMT).

> **Note**: The local time in London is the same as UTC time.

The following table lists the UTC time standards used in Unifier.

Code	Time Standard/Zone	Information	Type
10	UTC+13:00	Samoa	Pacific/Apia
20	UTC-12:00	International Date Line West	Etc/GMT+12
21	UTC-10:00	Hawaii	Pacific/Honolulu
30	UTC-11:00	Coordinated Universal Time-11	Etc/GMT+11
40	UTC-09:00	Alaska	America/Anchorage
41	UTC-07:00	Chihuahua, La Paz, Mazatlan	America/Chihuahua
50	UTC-08:00	Pacific Time (US and Canada)	America/Los_Angeles
51	UTC-08:00	Baja California	America/Santa_Isabel
70	UTC-07:00	Arizona	America/Phoenix
71	UTC-07:00	Mountain Time (US and Canada)	America/Denver
80	UTC-06:00	Central America	America/Guatemala
90	UTC-06:00	Central Time (US and Canada)	America/Chicago
91	UTC-06:00	Saskatchewan	America/Regina
92	UTC-06:00	Guadalajara, Mexico City, Monterrey	America/Mexico_City
110	UTC-05:00	Eastern Time (US and Canada)	America/New_York

Code	Time Standard/Zone	Information	Type
111	UTC-05:00	Indiana (East)	America/Indianapolis
112	UTC-05:00	Bogota, Lima, Quito	America/Bogota
130	UTC-04:00	Atlantic Time (Canada)	America/Halifax
131	UTC-04:30	Caracas	America/Caracas
132	UTC-04:00	Asuncion	America/Asuncion
133	UTC-04:00	Cuiaba	America/Cuiaba
134	UTC-04:00	Georgetown, La Paz, Manaus, San Juan	America/La_Paz
135	UTC-04:00	Santiago	America/Santiago
140	UTC-03:30	Newfoundland	America/St_Johns
150	UTC-03:00	Brasilia	America/Sao_Paulo
151	UTC-03:00	Buenos Aires	America/Buenos_Aires
152	UTC-03:00	Cayenne, Fortaleza	America/Cayenne
153	UTC-03:00	Greenland	America/Godthab
154	UTC-03:00	Montevideo	America/Montevideo
155	UTC-03:00	Salvador	America/Bahia
160	UTC-02:00	Coordinated Universal Time-02	Etc/GMT+2
170	UTC-01:00	Azores	Atlantic/Azores
171	UTC-01:00	Cape Verde Is.	Atlantic/Cape_Verde
180	UTC	Dublin, Edinburgh, Lisbon, London	Europe/London
181	UTC	Dublin	Europe/Dublin
182	UTC	Casablanca	Africa/Casablanca
183	UTC	Coordinated Universal Time	Etc/GMT
184	UTC	Monrovia, Reykjavik	Atlantic/Reykjavik
200	UTC+01:00	Amsterdam	Europe/Amsterdam
201	UTC+01:00	Brussels, Copenhagen, Madrid, Paris	Europe/Paris
202	UTC+01:00	Belgrade, Bratislava, Budapest, Ljubljana, Prague	Europe/Budapest
203	UTC+01:00	Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna	Europe/Berlin

Code	Time Standard/Zone	Information	Type
204	UTC+01:00	Sarajevo, Skopje, Warsaw, Zagreb	Europe/Warsaw
205	UTC+01:00	West Central Africa	Africa/Lagos
206	UTC+01:00	Windhoek	Africa/Windhoek
210	UTC+02:00	Athens, Bucharest	Europe/Bucharest
211	UTC+02:00	Cairo	Africa/Cairo
212	UTC+02:00	Beirut	Asia/Beirut
213	UTC+02:00	Jerusalem	Asia/Jerusalem
214	UTC+02:00	Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius	Europe/Helsinki
215	UTC+02:00	Harare, Pretoria	Africa/Johannesburg
220	UTC+02:00	Damascus	Asia/Damascus
221	UTC+02:00	Istanbul	Europe/Istanbul
222	UTC+02:00	Tripoli	Africa/Tripoli
223	UTC+03:00	Amman	Asia/Amman
224	UTC+03:00	Kaliningrad, Minsk	Europe/Kaliningrad
225	UTC+03:00	Kuwait, Riyadh	Asia/Riyadh
230	UTC+03:00	Nairobi	Africa/Nairobi
231	UTC+04:00	Moscow, St. Petersburg, Volgograd	Europe/Moscow
232	UTC+03:00	Baghdad	Asia/Baghdad
240	UTC+03:30	Tehran	Asia/Tehran
250	UTC+04:00	Abu Dhabi, Muscat	Asia/Dubai
251	UTC+04:00	Baku	Asia/Baku
252	UTC+04:00	Port Louis	Indian/Mauritius
253	UTC+04:00	Tbilisi	Asia/Tbilisi
254	UTC+04:00	Yerevan	Asia/Yerevan
260	UTC+04:30	Kabul	Asia/Kabul
261	UTC+05:00	Ashgabat, Tashkent	Asia/Tashkent
270	UTC+06:00	Yekaterinburg	Asia/Yekaterinburg
280	UTC+05:00	Islamabad, Karachi	Asia/Karachi
290	UTC+05:30	Chennai, Kolkata, Mumbai, New Delhi	Asia/Calcutta
300	UTC+06:00	Astana	Asia/Almaty

Code	Time Standard/Zone	Information	Type
301	UTC+06:00	Dhaka	Asia/Dhaka
302	UTC+06:30	Yangon (Rangoon)	Asia/Rangoon
310	UTC+07:00	Bangkok, Hanoi, Jakarta	Asia/Bangkok
311	UTC+08:00	Krasnoyarsk	Asia/Krasnoyarsk
320	UTC+08:00	Perth	Australia/Perth
321	UTC+08:00	Beijing, Chongqing, Hong Kong, Urumqi	Asia/Shanghai
322	UTC+08:00	Kuala Lumpur, Singapore	Asia/Singapore
323	UTC+08:00	Taipei	Asia/Taipei
324	UTC+09:00	Irkutsk	Asia/Irkutsk
340	UTC+09:00	Seoul	Asia/Seoul
341	UTC+09:00	Osaka, Sapporo, Tokyo	Asia/Tokyo
342	UTC+10:00	Yakutsk	Asia/Yakutsk
350	UTC+09:30	Darwin	Australia/Darwin
351	UTC+09:30	Adelaide	Australia/Adelaide
360	UTC+10:00	Brisbane	Australia/Brisbane
361	UTC+10:00	Canberra, Melbourne, Sydney	Australia/Sydney
362	UTC+10:00	Hobart	Australia/Hobart
363	UTC+11:00	Vladivostok	Asia/Vladivostok
364	UTC+10:00	Guam, Port Moresby	Pacific/Port_Moresby
380	UTC+07:00	Novosibirsk	Asia/Novosibirsk
390	UTC+08:00	Ulaanbaatar	Asia/Ulaanbaatar
391	UTC+11:00	Solomon Is., New Caledonia	Pacific/Guadalcanal
392	UTC+12:00	Magadan	Asia/Magadan
400	UTC+12:00	Coordinated Universal Time+12	Etc/GMT-12
410	UTC+12:00	Fiji	Pacific/Fiji
411	UTC+12:00	Auckland, Wellington	Antarctica/McMurdo
420	UTC+13:00	Nuku'alofa	Pacific/Tongatapu
430	UTC+05:30	Sri Jayawardenepura	Asia/Colombo
440	UTC+05:45	Kathmandu	Asia/Katmandu

Date Format

The date format is a user preference. You must enter a value for the user. The date format is a user preference setting that the user can change later if necessary. The date format can also be set in a user preference template.

The following lists the supported date formats:

Note: The time format for all dates is: `HH:MM AM`.

Date Format	Code
MM/DD/YYYY	0
DD/MM/YYYY	1
MM/DD/YY	2
DD/MM/YY	3
MM-DD-YYYY	4
DD-MM-YYYY	5
MM-DD-YY	6
DD-MM-YY	7
DD.MM.YYYY	8
YYYY-MM-DD	9
MMM/DD/YYYY	10
DD/MMM/YYYY	11
YYYY/MMM/DD	12
M/D/YYYY	13
M/D/YY	14
D/M/YYYY	15
D/M/YY	16

Date Format	Code
YY/M/D	17
YYYY/M/D	18
YY/MM/DD	19
YYYY/MM/DD	20

User Type

Use the values included in the following table for any record in which Standard or Portal are the User Type choices.

The User Type you choose, for the import value, will be used as soon as the record is imported.

For example, if you enter a “0” for the User Type, the user will be a Standard Unifier user upon successful import of the record; if you enter “1”, the user will be a Portal user.

User Type	Value
Standard	0
Portal	1

Company Address

Use the codes included in the following table for the address fields, when entering the user records.

Address Type	Code
Headquarters	1
Main	2
Branch Office	3
Billing	4
Shipping	5
Billing and Shipping	6
Satellite Office	7

Status

Use the values included in the following table for any record in which Active, Inactive, or On Hold are the status choices. The status you choose for the import value will be effective as soon as the record is imported. For example, if you enter a “1” for the status, the user will be an active Unifier user upon successful import of the record; if you enter “2”, the user will be On Hold until an administrator activates the user record from within Unifier.

Status	Value
Inactive	0
Active	1
On Hold	2

Check Box

For check boxes on BP forms, you can import the following values to select (“check”) a check box or leave it unchecked.

Check Box	Value
Check box not selected	0
Check box selected	1

Return Values

For all of the following methods, the return object will be XMLObject.

> **Note**: XMLObject is described at the beginning of this document.

XMLObject contains 3 string elements:

- **String Status Code** (see table below)
- **String Array** – error log indicating reason of failure
- **String XMLContents** – a string contains XML values, if a return value is expected
- **XMLFileObject** – a file object that contains files as return value

The following is the list of String status codes and their descriptions:

String Status Code	Description
200	OK
300	Bad Data
302	Invalid Short name/AuthCode
303	Invalid Project Number
304	Invalid Business Process Name
405	Method Not Allowed
406	Invalid Server URL
500	Server Error, contact the system administrator
501	Not Implemented
503	Out of Resources
505	Invalid data
506	Invalid login
507	Error in creating directory for business process
601	Invalid Shortname/Authentication code
602	Project Number is not Correct

String Status Code	Description
603	Business Process Name is not correct
605	No Budget setup for the project
606	Budget is not correct.
608	XML is not valid
609	Project Template/Project Number is not correct
610	Only Cost and Line Item BPs are supported
611	Only Cost and Line Item BPs are supported
612	Project Template/Project name already exists
613	Project number already exists
614	ProjectXML should contain project number and project name tags.
615	Error processing XML.
616	XML is null or empty.
617	Found duplicate project numbers.
650	Create BP failed because …..
651	CBS Code is not Present
654	CBS Code is not in the correct format.
655	Could not create CBS Item.
656	Object Name is null or empty.
657	Invalid record_no.
658	field record_no is not present.
659	field record_no is empty.
660	field record_no is duplicate.
661	Integration settings for default workflow link id not set.
662	Integration setting for default workflow or creator or BP not set. The Integration setting for default workflow, creator, or Business Process (BP) name has not been set. This error is sent for the Project number and the BP name that the user has sent in the web service call.
663	Status field value is not correct.
664	Create Project failed because

String Status Code	Description
665	Data values for field tag
690	Only Lineitem non workflow BPs supported
691	Rule engine.
692	Cannot create multiple records for single record BP
693	Cannot create workflow process
694	Project Name can only contain the following characters: alphanumeric, spaces, and ()[],.;:-_&<>'
695	Project Number can only contain the following characters : alphanumeric and [.-_] characters are allowed
701	Invalid Column Name.
702	Column is not defined as manual entry.
703	Administrator is inactive. Cannot create column data.
704	Duplicate Column Name.
705	Update Column Data failed because:
706	Get Column Data failed because:
707	Cost sheet does not exist.
708	Unable to run report. Contact System Administrator.
709	Report name is not valid. Check if report exists or is enabled for Integration.
710	Cannot create Asset type BP record using createBPRecord interface.
711	Invalid sheet name.
712	Import of Schedule activities failed reason :
713	Schedule sheet name is not valid.
714	Schedule sheet cannot be created; reason XML Import property is set to 'merge into existing schedule'.
715	Schedule sheet cannot be updated, reason XML Import property is set to 'overwrite existing schedule'.
716	The SOV of the selected contract is locked by another Business Process.
717	Selected contract does not have SOV. Check if the contract has reached terminal status.
718	Field name "+bp_picker_source_name +" is required.

String Status Code	Description
719	pickervalue +" is invalid "+"value for "+source_name +"."
720	Reference BP not supported
721	Multiple records found for field "+source_name + " with value " + pickervalue+ "."
722	Invalid field name(s)
723	Spend Business Process not supported.
724	SOV is not created yet for the base commit associated with this Business Process.
725	You cannot change a Business Process picker name + <source_name> + , that has line item(s) associated with it.
726	Reference Commit picker value cannot be changed.
800	No Active multiple instance
803	Copy from shell template is not active
804	(can be a combination of the following given below) can only contain the following characters: alphanumeric and [.-_] characters cannot contain any of the following characters: / \ : * ? " < > | Current shell cannot be a sub-shell of the selected shell Administrator is not a valid user
805	shell_name <shell name> already exists under <location>
806	shell_number <shell num> already exists
807	shell_name <shell name> already exists under <location>
808	shell_number <shell num> already exists
810	Integration Interface not found
811	Invalid Filter Condition
812	Copy from shell/template number is not correct
854	Amount exceeds remaining Reference Commit balance for one or more CBS codes
855	Reference commit value cannot be modified

String Status Code	Description
867	Correct the syntax and resubmit the statement. The Data Type is not supported. Data Type should be one of the following: ▸ TIMESTAMP ▸ INTEGER ▸ FLOAT ▸ VARCHAR ▸ CLOB ▸ XML
869	Number of line items in the input file exceeds number of lines in the SOV

Copyright

Oracle Primavera Unifier Integration Interface Guide