User Guide:

Adapt V11 Web Services

V2 Interfaces

Adapt V11 Web Services
V2 Interfaces

Date April 2nd, 2020
Version Version 1.0

Document History

Version	Date	Author	Summary
1.0	April 2nd, 2020	Rob Hayesmore	Document created from V1 webservices guide, version 4.93

Table of Contents

Glossary and Abbreviations

Term or Abbreviation	Definition
JWSDP	Java Web Service Development Pack
Axis	Apache Jakarta Axis – SOAP java library
BO	Adapt V11 Business Object
Entity	A "record" within the system. Most of system objects (like Candidate, Job and Client) are stored as Entities.
Role	An entity type (e.g. Perm Candidate or Client). Defines the Entity properties and how the system will treat this Entity.
Properties	A collection of attributes that are logically connected in some way (e.g. Address). Represented by a table in the underlying DB.
Attribute	An individual data item or field (e.g. Post Code). Represented by a column in the underlying DB.
Single Occurrence Property	Only one instance of the property exists per entity.
Named Occurrence Property	A fixed number of occurrences of the property exist per entity. Each occurrence is addressed by name. For example, the Address property has two named occurrences: Primary and Secondary.
Multiply Occurring Property	An unlimited number of occurrences per entity. For example, the Skills and Job Category properties are multiply occurring properties.
SOAP	Simple Object Access Protocol – a standard method for accessing and describing an API over HTTP(S).
XML	eXtensible Markup Language - a metalanguage which allows users to define their own customized markup languages
Web Services	An API accessible over internet technologies
API	Application Programming Interface – an interface allowing external applications to access/control and application.

1. Overview

The Adapt SOAP Web Services are a collection of XML-based Web Services using the SOAP protocol that allows developers to integrate most of Adapt V11 functionality into their applications and business processes. The Web Services provide an extensive set of Adapt V11 system data and functionality, such as executing Business Objects, searching and accessing Entities, Documents, Tasks, Journal entries, Calendar Appointments, Code Groups, and other types of data that can be used within applications.

For example, Adapt Web Services can be used to:

- Execute a Business Object using the BOExecutor service, passing in data and receiving a response.
- Run a pre-defined Query via the Search service and retrieve the result set.
- Create, update or delete entities using the Entity service.

2. Getting Started

The Adapt Web Services are made up of eleven SOAP services accessible via their own unique endpoint URLs, each with a WSDL that describes the methods and objects available in the corresponding service. The eleven services are:

1. Authentication
2. BO Executor
3. Search
4. Entity
5. Document
6. Code Group
7. Tasks
8. Journal
9. Calendar
10. Meta Data
11. Data Admin

2.1 Accessing the Services and WSDLs

The URLs to access the Web Services share a common structure:

`https://<server:port>/webservices/<servicename>`

The `<server:port>` element of the URL is client dependent, but well known as it is the same as used by users to connect to Adapt. For example, at the time of writing, clients using the hosted version of Adapt were split over 4 clusters, each with their own server:

1. Cluster 1: c1.adaptondemand.com
2. Cluster 2: c2.adaptondemand.com
3. Cluster 3: c3.adaptondemand.com
4. Cluster 4: c4.adaptondemand.com

The `<servicename>` element can be found from the table below:

Service	Service Name
Authentication	LogonServiceV2
BO Executor	BOExecServiceV2
Search	SearchServiceV2
Entity	EntityServiceV2
Document	DocServiceV2
Code Group	CodeServiceV2
Tasks	TaskServiceV2
Journal	JournalServiceV2
Calendar	CalServiceV2
Meta Data	MetaDataServiceV2
Data Admin	ManageRequestServiceV2

For example, access to the Authentication service for a client hosted on cluster 2 would be achieved through the URL:

`https://c2.adaptondemand.com/webservices/LogonServiceV2`

The WSDL for any service can be retrieved by appending `?wsdl` to the appropriate URL. For example, retrieve the WSDL for Authentication methods from the URL:

```
https://c2.adaptondemand.com/webservices/LogonServiceV2?wsdl
```

Many development environments or languages offer facilities to build interface classes automatically from WSDLs, or to dynamically use a WSDL at the time of execution.

2.2 Authentication

The majority of the Adapt Web Service methods require an access token to be provided as one of the parameters. This is referred to in the documentation as the session ID or SID. A SID is obtained by making a call to the `logon()` method of the Authentication service.

Passing an incorrect, invalid or expired SID to a method will result in failure of that method to execute.

SIDs expire after approximately 3 minutes of non-usage, at which point a new SID should be obtained by making a new call to the `logon()` method. If your code chooses to cache a SID (and this is not recommended) it will need to detect an expiry failure and re-request a SID.

For this reason it is recommended that an atomic set of calls to the Web Services (for example, to obtain data and serve a web page) include a call to the `logon()` method to obtain a SID at the beginning, use the SID to execute the Web Service call(s) needed to complete the operation and then use the `logoff()` method to release the SID.

2.3 Exceptions

The Adapt Web Services adhere to the SOAP standard for throwing exceptions, all exceptions will be wrapped into a SOAPFault object (returned in the `<fault>` section of the response). This should allow the development environment/language to correctly handle and pass on the exception to the calling code.

The following types of exceptions are defined in Adapt V11 Web Services and can be thrown via the standard SOAPFault. These four exceptions are exposed through the WSDLs:

- **InvalidArgumentException** - thrown if arguments passed to web service method are invalid (null objects, empty strings etc.);
- **DataNotFoundException** - thrown if no data can be found matching method arguments;
- **AccessDeniedException** – thrown if there is no access to a service. This could be because:
    - there is no permission in domain profile for requested service;
    - session ID is invalid;
    - incorrect login/password, etc.
- **ServerErrorException** – thrown if server encounters any error while executing service method not covered by the above exceptions.
- **InvalidCodeException** – thrown if the code id passed to the webservice method is invalid or retrieved code object is null
- **InvalidCodeGroupException** – thrown if the code group id passed to the webservice method is invalid or codegroup object is null

2.4 Ownership

The Adapt application supports data visibility control based on a hierarchical group structure, sometimes referred to as data segregation. Typically, it is used to control visibility between different teams or branches of a recruitment agency.

The Web Services respect this visibility control and will throw a **DataNotFoundException** if an attempt is made to update or retrieve an entity that the user does not have access to. In the case of the Web Services the user that is used for authentication is the one that must have access to entities.

2.5 Time Zones

AdaptService allows you to work in different Time Zones to perform correct operations with **Calendar** and **Task** services. You can provide your Time Zone to the service and expect it to work in it. However if the Time Zone provided is not supported by the server, **InvalidArgumentException** will be thrown. If the Time Zone wasn't provided, service will work in the server's default time zone.

Time Zones are provided by their names, e.g. GMT, GMT0, Greenwich, UTC, Universal, Zulu, Europe/Amsterdam, Europe/Berlin.

2.6 Locales

AdaptService works with different Locales. Locale provided by the user is checked against locales supported by the server (based on available languages in Adapt V11 Configuration Domain used by the service). If locale provided isn't valid, InvalidArgumentException will be thrown.

Locales are provided by their names, e.g. EN, EN_US, RU_RU, ES_ES

3. API Reference

This section includes topics to help you develop applications using the AdaptService Web Service.

3.1 Authentication Service

The Authentication Service is used to initiate and close web service session logon. It also provides supplementary functions used to retrieve versioning information and information about time zones and locales supported by the server.

3.1.1 Summary of Methods

Method	Purpose
logon	starts new web services session. Accepts username, password, domain profile, domain, [locale, timezone, dateformat, timeformat] and returns long session id, required to call any other method.
logoff	closes existing session by id.
getAvailableTimezones	returns array of strings with all time zones recognizable by server.
getAvailableLocales	returns array of strings with all locales recognizable by server
getVersion	returns VersionInfo object with web service version information
getAvailableLanguageName	returns array of LanguageBean objects which contains all active language information.

3.1.2 Objects

NONE

3.1.3 Logon

Use the logon method to establish a new web service session and obtain a session ID to be used with the other services and methods. This will generally be the first call made to the Adapt Web Services.

```
long logon(String String_1,
           String String_2,
           String String_3,
           String String_4,
           String String_5,
           String String_6,
           int    int_7,
           int    int_8)
```

Method arguments:

Argument	Type	Description
String_1	String	Username
String_2	String	Password
String_3	String	Domain Name
String_4	String	Domain Profile Name
String_5	String	Locale name, one of those retrieved by getAvailableLocales(). For example: EN, EN_US, RU_RU, ES_ES_Traditional
String_6	String	Time Zone name. Time zones supported by Adapt can be retrieved by calling getAvailableTimezones ().

int_7	Integer	Date format. Allowed values are: • 0 – Default • 1 – Long • 2 – Medium • 3 – Short
int_8	Integer	Time format. Allowed values are: • 0 - Full • 1 - Long • 2 - Medium • 3 - Short

Return value:

Type	Description
Long	Id of new session.

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.1.4 Logoff

Closes existing session by id

```
void logoff(long long_1)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID (SID)

Return value: N/A

Exceptions:

- DataNotFoundException
- ServerErrorException

3.1.5 getAvailableTimezones

Returns array of strings with all time zones server recognises. Does not require a SID to be called allowing time zone information to be collected prior to authentication.

```
String[] getAvailableTimezones();
```

Method arguments: N/A

Return value:

Type	Description
String[]	All time zones server recognizes.

Exceptions: N/A

3.1.6 getAvailableLocales

Returns array of strings with all locales server recognizes. Does not require a SID to be called allowing time zone information to be collected prior to authentication..

```
String[] getAvailableLocales(String String_1)
```

Method arguments:

Argument	Type	Description
String_1	String	Domain Name

Return value:

Type	Description
String[]	All locales recognizable by server.

Exceptions:

- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.1.7 getVersion

Returns VersionInfo object with web service version information

```
VersionInfo getVersion();
```

Method arguments: N/A

Return value:

Type	Description
VersionInfo	Holds service and endpoints version.

Exceptions: N/A

3.1.8 getAvailableLanguageNames

Returns array of LanguageBean which contains all active language information

```
LanguageBean[]getAvailableLanguageNames (String String_1)
```

Method arguments:

Argument	Type	Description
String_1	String	Domain Name

Return value:

Type	Description
LanguageBean[]	All active languages information.

Exceptions:

- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.2 Business Object Executor Service

One can run Adapt V11 Business Objects using IBOExecutorV1 interface. Interface consists of one method called executeBO.

If an Entity is created via executeBO method, it gets associated in the ENTITY_OWNERSHIP table not only with the user who executed a BO (via web services) but also with other users that share user groups with him.

3.2.1 Summary of Methods

Method	Purpose
executeBO	Used to execute a Business Object (BO) optionally passing data into it and retrieving a response.

3.2.2 Objects

Control Value Object

Member Name	Type	Description
name	String	Can be **ENTITYID** to store value of the current entity or other client specific name.
controlPath	String	Path of the view control element.
value	String	Control value
dataType	String	Control data type, e.g. NUMERIC, DATE, TIME

3.2.3 executeBO

Runs BO with given string data, preferably XML and returns result as string, preferably XML. Method accepts session id, BO name, and xml string. Returns XML generated by BO

```
String executeBO(long           long_1,
                 String         String_2,
                 String         String_3,
                 ControlValue[] arrayOfControlValue_4)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Name of Business Object
String_3	String	String value passed to Business Object – typically XML or JSON to be used by the Business Object
arrayOfControlValue_4	ControlValue[]	Array of control values that may be used by BO

Return value:

Type	Description
String	String value returned by Business Object – typically XML or JSON but can be any string.

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.3 Search Service

Search Interface can be used to run queries previously created in Query Explorer tool. You may also access those queries and saved queries search results.

3.3.1 Summary of Methods

Method	Purpose
getAllQueriesNames	Retrieve a list of Queries available to the user
runQuery	Execute a query and return list of matching entities. Optional save result set for future access.
getSearchResults	Retrieve list of saved result sets
getSearchResult	Retrieve contents of a saved result set
getSearchResultXML	Retrieve field level data based on a saved result set.
getSearchResultCount	Retrieve number of entities in a saved result set
deleteSearchResult	Delete a saved result set
getSearchResultWithRoles	Retrieve contents of a saved result set (like getSearchResult) with additional information defining the default role of the entities included in the result set. Avoids having to fetch each entity to determine its role.
quickFind	Find entities using the quickfindable attributes defined in Adapt

3.3.2 Objects

SearchParameter Object

The search parameter object defines an individual parameter that is substituted into the query logic for matching. When executing a query an array of search parameters is passed in.

Name	Type	Description
dataType	Integer	Data type of the passed parameter. • 1 – String • 2 – Numeric • 3 – Date • 4 – Character
name	String	Name of the query parameter
stringValue	String	Holds the value of the parameter for a string type parameter
longValue	Long	Holds the value of the parameter for a numeric type parameter
dateValue	Date	Holds the value of the parameter for a date type parameter

SearchResultV1 Object

Name	Type	Description
ID	long	ID of the result set
createdBy	String	Name of user that created the result set
createdByID	long	User ID of user that created the result set
createdDate	Date	Date result set was created
foundEntities	long[]	List of entity IDs from the result set
highlightColor	int	Highlight colour ID
label	String	Name of the result set

modifiedBy	String	Name of user that last modified the result set
modifiedByID	long	Use ID of user that last modified the result set
modifiedDate	Date	Date result set was last modified
owner	String	Owner Type
ownerID	long	Owner ID
queryName	String	Name of query that was run to generate the result set

SearchResultWithRoles Object

Name	Type	Description
ID	long	ID of the result set
createdBy	String	Name of user that created the result set
createdByID	long	User ID of user that created the result set
createdDate	Date	Date result set was created
foundEntities	long[]	List of entity IDs from the result set
highlightColor	int	Highlight colour ID
label	String	Name of the result set
modifiedBy	String	Name of user that last modified the result set
modifiedByID	long	Use ID of user that last modified the result set
modifiedDate	Date	Date result set was last modified
owner	String	Owner Type
ownerID	long	Owner ID
queryName	String	Name of query that was run to generate the result set
entitiesRoles	EntityRoleBean[]	List of entity IDs from result set with corresponding default Role (use instead of **foundEntities** array if default role information is needed).

EntityRoleBean Object

Member Name	Type	Description
defaultRoleName	String	Name of the default role for the entity
entityId	long	ID of the entity

ColumnFormat Object

Name	Type	Description
referencePath	String	See description below
language	String	Name of any language registered in Adapt (e.g. English, French, German). This field is applicable for code attributes only and used to retrieve description in necessary language.
display	Short	Applicable for code attributes only. Possible values are: 0 – returning raw code value, 1 – code names are returned, 2 – returning code descriptions.
sort	Short	Possible values of sort order are: Attribute Data Type Value — Non CODE — CODE 0 — no ordering — no ordering

		1	ascending by display value	ascending by Code ID
		2	descending by display value	descending by Code ID
		3	ascending by display value	ascending by Code Name
		4	descending by display value	descending by Code Name
		5	ascending by display value	ascending by Code Description
		6	descending by display value	descending by Code Description

The **referencePath** field contains sequence of reference strings. At least one reference should be specified.

Each reference consists of property name and 1-2 attribute names: optional reference attribute and mandatory value attribute.

The property name is the name of either system (like ENTITY_TABLE) or ordinary property (including "X_" tables) without prefix "PROP_" and could include optional occurrence name for named occurrence property.

The reference attribute is used to link current property with preceding property or with the root table.

The value attribute is used to retrieve reference value and in case of subsequent references to link with following property.

The CREATEDDATE, CREATED_BY, UPDATEDDATE value attribute names are allowed for the ENTITY_TABLE system property.

The USER_ID, USER_NAME, INITIALS, LOGIN_NAME, EMAIL value attribute names are allowed for U_PERSONAL table.

The format of **referencePath** field of **ColumnFormat** is as follows:

```
REF_ATTR_1.PROP_1(OCC_OF_PROP_1).ATTR_1:REF_ATTR_2. PROP_2(OCC_OF_PROP_2).ATTR_2:REF_ATTR_3.
PROP_3(OCC_OF_PROP_3)…
```

Where `REF_ATTR_1.PROP_1(OCC_OF_PROP_1).ATTR_1` is the first reference and `REF_ATTR_2. PROP_2(OCC_OF_PROP_2).ATTR_2` is the second one and so on.

The `REF_ATTR_1` is the name of the reference attribute of the first reference.

The `PROP_1` is the name of the property.

The `OCC_OF_PROP_1` is the name of the occurrence for the first reference property.

The `ATTR_1` is the value attribute name.

The given sequence of references means that for each item in the result the instance of PROP_1 will be found where ENTITY_ID of the result item equals to REF_ATTR_1 value.

And for each PROP_1, instance of PROP_2 will be found, where value of ATTR_1 equals to REF_ATTR_2 and so on.

The reference attribute name can be omitted. In this case attribute predefined in Adapt will be used.

The last value attribute is column value. The sorting and display format is applied to it.

Examples of **referencePath**:

```
PERSON_GEN.FULLNAME
ADDRESS(Primary).COUNTRY
ENTITY_TABLE.UPDATEDDATE
X_ASSIG_CAND.JOB:JOB_GEN.JOB_TITLE
OFFICE.OWN_OFFICE.REFERENCE:REFERENCE.CLIENT_GEN.NAME
REFERENCE.OWN_CONS(Permanent).CONSULTANT:USER_ID.U_PERSONAL.USER_NAME
```

3.3.3 getAllQueriesNames

This method returns array of names of all queries available to current user.

```
String[] getAllQueriesNames( long long_1 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id

Return value:

Type	Description
String[]	Array of names of all queries available to current user

Exceptions:

- AccessDeniedException
- ServerErrorException
- DataNotFoundException

3.3.4 runQuery

This method executes a query and returns the Search Result object (SearchResultV1) containing array of found entities and all supplementary information. Accepts array of search parameters (SearchParameter) and can save or not save search results to Adapt V11 database depending on saveResults boolean parameter. This method accepts firstItemIndex and itemCount parameters to limit size of returning data, to prevent slowdown when retrieving results with a big number of found entities.

If Business Object is associated with a query in the Query Explorer Tool, it will be executed after the execution of the Query. The parameters which are not used by query will be available to Business Object.

```
SearchResultV1 runQuery ( long              long_1,
                          String            String_2,
                          SearchParameter[] arrayOfSearchParameter_3,
                          int               int_4,
                          int               int_5,
                          boolean           boolean_6,
                          String            String_7 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Name of the query
arrayOfSearchParameter_3	SearchParameter[]	Parameters of the search
int_4	Integer	Index of the first item
int_5	Integer	Maximum number of items to be returned
boolean_6	Boolean	Should results be saved or not
String_7	String	User name to assign results to. If not provided, results are assigned to service user.

Return value:

Type	Description
SearchResultV1	Search result data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

Note: The method runQuery() executes the configured pre-query BO before executing the query. The execution of the query depends upon the status returned by the pre-query BO, as given below:

1. No pre-query BO is configured for a query: The query executes as normal.
2. The pre-query BO returns a status which is greater than 0: The query is not executed
3. The pre-query BO returns a status of -1: The query is not executed
4. A system error occurs while executing the pre-query BO: The query is not executed
5. The pre-query BO returns a status of 0: The query executes as normal.

3.3.5 getSearchResults

This method returns array of IDs of all stored query results sets that match the filter criteria. Filter is specified via array of search parameters (SearchParameter, see **3.3.2.1**).

```
long[] getSearchResults(long              long_1,
                        SearchParameter[] arrayOfSearchParameter_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
arrayOfSearchParameter_2	SearchParameter[]	Filters. Array of search parameters defining the filter(s) to be applied. Only saved result sets matching the parameters will be returned.

Return value:

Type	Description
Long[]	Array of IDs of all stored query result sets that match the filter criteria

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

Filters

To filter the list of result sets returned, supply filter criteria as an array of Search Parameter objects. The objects in the array can be one or more of the following allowed values.

To filter by	SearchParameter member details to use		
	name	dataType	Value goes in
The title assigned to the query result	**Label**	1	stringValue
user ID of the query result creator	**Created By**	2	longValue
Date range when query result was created	**Created Start Date**	3	dateValue
	Created End Date	3	dateValue
User ID of the last user to modify the result set	**Modified By**	2	longValue
Date range when query result was modified	**Modified Start Date**	3	dateValue
	Modified End Date	3	dateValue
Method of the query invocation (e.g. B – Business Object, Q - query)	**Method**	4	stringValue
User ID of user that highlighted result set	**Highlight**	2	longValue

3.3.6 getSearchResult

This method returns stored query result in form of Search Result object (SearchResultV1) according to the formatter object of the search result/user/domain. This method also accepts firstItemIndex and itemCount parameters to support paging.

```
SearchResultV1 getSearchResult( long long_1,
                                long long_2,
                                int  int_3,
                                int  int_4 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Saved search result ID
int_3	Integer	Index of the first item
int_4	Integer	Maximum number of items to be returned

Return value:

Type	Description
SearchResultV1	stored search result

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.3.7 getSearchResultXML

This method returns stored query result in form of XML string with formatter settings obtained by search result (not search definition) id. Method accepts array of ColumnFormat objects in order to provide convenient formatting of columns in returning XML data set.

```
String getSearchResultXML( long           long_1,
                           long           long_2,
                           int            int_3,
                           int            int_4,
                           ColumnFormat[] arrayOfColumnFormat_5 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Saved search result id
int_3	Integer	Index of the first item
int_4	Integer	Maximum number of items to be returned
arrayOfColumnFormat_5	ColumnFormat[]	Array of ColumnFormat objects defining the column data to be returned.

Return value:

Type	Description
String	XML of saved search result

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException

- ServerErrorException

Search Result XML

XML of saved search result has the following format:

```
<SearchResult searchID='SEARCH_ID'>
  <Entity id='ENTITY_ID' defaultrole="DEFAULT_ROLE" resultIndex='1'/>
        …
  <Entity id='ENTITY_ID' defaultrole="DEFAULT_ROLE" resultIndex='N'/>
</SearchResult>
```

3.3.8 getSearchResultCount

This method returns number of records in specified stored query result.

```
long getSearchResultEntityCount( long long_1,
                                 long long_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Saved search result id

Return value:

Type	Description
Long	Number of records in query result

Exceptions:

- AccessDeniedException
- IDataNotFoundException
- ServerErrorException

3.3.9 deleteSearchResult

This method permanently deletes search result by specified id

```
void deleteSearchResult( long long_1,
                         long long_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Saved search result id

Return value: none

Exceptions:

- AccessDeniedException
- IDataNotFoundException
- ServerErrorException

3.3.10 getSearchResultWithRoles

This method returns stored query result in form of Search Result object (SearchResultWithRoles). This method is similar to the getSearchResult method except returning value (SearchResultWithRoles instead of SearchResultV1). The SearchResultWithRoles object extends the SearchResultV1 by adding new field - array of 'EntityRoleBean' beans, which in their turn contain an entity Id and its default role name. This method also accepts firstItemIndex and itemCount parameters to limit size of returning data.

```
SearchResultWithRoles getSearchResultWithRoles( long long_1,
                                                long long_2,
                                                int  int_1,
                                                int  int_2 )
```

Method arguments:

Argument	Type	Usage	Description
long_1	Long	Manadatory	Session id
long_2	Long	Manadatory	Saved search result ID
int_1	Integer	Manadatory	Number of the Page to represent found entries from
int_2	Integer	Manadatory	Maximum number of items to be returned

Return value:

Type	Description
SearchResultWithRoles	stored search result

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.3.11 quickFind

This method performs search and returns its result in form of XML string. All method arguments are mandatory. Search results are not limited. Search criteria is taken from the attributeToCompare – it must be a quickfindable attribute and hence it has the criteria defined.

```
String quickFind( long   long_1,
```

```
String String_2,
String String_3,
String String_4,
String String_5 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Config name of the property
String_3	String	Config name of the attribute to be searched on. Must be quickfindable attribute
String_4	String	Config name of the attribute whose value will be returned
String_5	String	The value to look for in the **attributeToCompare**

Return value:

Type	Description
String	XML result of performed search.

Exceptions:

- AccessDeniedException
- InvalidArgumentException – when a non-quickfindable attribute is supplied
- DataNotFoundException
- ServerErrorException

XML of performed search has the following format:

```
<SearchResult >
  <Entity id='id' defaultrole='DEFAULT_ROLE'>
    <Property name='PROPERTY_NAME'>
      <Attribute name='ATTRIBUTE_NAME'>
        Attribute value
      </Attribute>
    </Property>
  </Entity>
  ...
</SearchResult>
```

3.4 Entity Service

Entity Interface can be used to retrieve, create, update and delete Entities. It is also possible to retrieve User as Entity.

Entities can be retrieved and edited with regard to the user account used to authenticate the connection, so that the user can only view and edit Entities which have been created by him personally or by other users who share user groups with him.

The table below shows dependency between Entity Access Service methods and users with different access rights:

		Disable Entity Visibility = YES			Disable Entity Visibility = NO		
		Entity owned by User Group which is User's Full Access Group	Entity owned by User Group which is User's View Only Group	Entity owned by User Group which is neither User's Full Access Group nor View Only	Entity owned by User Group which is User's Full Access Group	Entity owned by User Group which is User's View Only Group	Entity owned by User Group which is neither User's Full Access Group nor View Only
V1							
getUser		+	+	+	+	+	+
getEntity		+	+	+	+	+	x
getEntityData		+	+	+	+	+	x
getEntityDataNoPrimaryReference		+	+	+	+	+	x
saveEntity	create new	+	+	+	+	+	+
saveEntityData	create new	+	+	+	+	+	+
saveEntity	update existing	+	x	x	+	x	x
saveEntityData	update existing	+	x	x	+	x	x
deleteEntity		+	x	x	+	x	x

3.4.1 Summary of Methods

Method	Purpose
getUser	Retrieve user details from the user table
getEntity	Get XML representing an entire entity
getEntityData	Get specified elements of an entity as XML
saveEntity	Update/Create an entity
saveEntityData	Update/Create an entity from a data fragment
getFavourtiteEntities	Retrieve a list if favourited entities
getEntityDataNoPrimaryReference	Get specified elements of an entity as XML without system attributes (reference and occurrence ID).

3.4.2 Objects

SearchParameter Object

The search parameter object defines a parameter used to identify a User.

Name	Type	Description
dataType	Integer	Data type of the passed parameter. • 1 – String • 2 – Numeric • 3 – Date

		• 4 – Character
name	String	Name of the query parameter
stringValue	String	Holds the value of the parameter for a string type parameter
longValue	Long	Holds the value of the parameter for a numeric type parameter
dateValue	Date	Holds the value of the parameter for a date type parameter

3.4.3 getUser

This method returns user data in form of XML, based on Adapt V11 core "get user as entity" functionality.

```
String getUser( long            long_1,
                SearchParameter SearchParameter_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
SearchParameter_2	SearchParameter	Search Parameter Object

Return value:

Type	Description
String	User XML

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

Search Parameter

Use the SearchParameter object to provide the criteria to locate the user record to be retrieved.

To search by	SearchParameter member details to use		
	name	dataType	Value goes in
User Name	**username**	1	stringValue
User ID	**userID**	2	longValue
Active Directory User Name	**ADName**	1	stringValue

3.4.4 getEntity

This method returns Entity XML by entity ID. Formatting rules like time zone, locale, and date and time format are specified at logon

```
String getEntity( long long_1,
                  long long_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Entity ID

Return value:

Type	Description
String	Entity XML

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

Entity XML format

getEntity method returns entity in XML representation. Format for this representation is as follows:

```
<Entity defaultrole='DEFAULT_ROLE'>
  <Role>Role1</Role>
  <Role>Role2</Role>
  <Role>RoleN</Role>
  <Property name=’PROPERTY_NAME_1’ occurrence=’OCCURRENCE_NAME’>
    <Attribute name=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
    <Attribute name=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
  </Property>
  <Property name=’PROPERTY_NAME_N’>
    <Attribute name=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
    <Attribute name=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
  </Property>
</Entity>
```

If referenced attribute in list of property is an entity, it will have the following format:

```
<Property name=’PROPERTY_NAME_1’ occurrence=’OCCURRENCE_NAME’>
  <Attribute name=’ATTRIBUTE_NAME’ defaultrole='DEFAULT_ROLE' sqldatatype=’ATTRIBUTE_TYPE’>
    Value
  </Attribute>
</Property>
```

To create a new *Entity,* an XML representation of the entity to be created must be constructed. The format of the XML for *Entity* creation is as follows:

```
<Entity defaultrole='DEFAULT_ROLE'>
  <Role>Role1</Role>
  <Role>Role2</Role>
  <Role>RoleN</Role>
  <Property name=’PROPERTY_NAME_1’ occurrence=’OCCURRENCE_NAME’>
    <Attributename=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
  </Property>
  <Property name=’PROPERTY_NAME_N’>
    <Attributename=’ATTRIBUTE_NAME’ sqldatatype=’ATTRIBUTE_TYPE’>Value</Attribute>
  </Property>
</Entity>
```

Where **’OCCURRENCE_NAME’** is occurrence from the *Named Property*.
Note that **’OCCURRENCE_NAME’** only needs to be specified for *Named Properties*

Below is an example of the XML that creates permanent candidate *Entity*.

```
<Entity defaultrole='PERM_CAND'>
    <Role>PERM_CAND</Role>
    <Property name='PERSON_GEN'>
        <Attribute name='MIDDLE_NAME' sqldatatype='NVARCHAR'>Middle</Attribute>
        <Attribute name='FIRST_NAME' sqldatatype='NVARCHAR'>First</Attribute>
        <Attribute name='LAST_NAME' sqldatatype='NVARCHAR'>Last</Attribute>
        <Attribute name='FULLNAME' sqldatatype='NVARCHAR'>Mr Phillips</Attribute>
        <Attribute name='TITLE' sqldatatype='NUMERIC'>1303812</Attribute>
        <Attribute name='GENDER' sqldatatype='NUMERIC'>1303874</Attribute>
  </Property>
  <Property name='CAND_GEN'>
    <Attribute name='PLACE_OF_B' sqldatatype='NVARCHAR'></Attribute>
    <Attribute name='NATIONAL_NUM' sqldatatype='NVARCHAR'></Attribute>
    <Attribute name='ID_NO' sqldatatype='NVARCHAR'></Attribute>
    <Attribute name='BENEFIT_FROM' sqldatatype='CHAR'>N</Attribute>
    <Attribute name='RIGHT_BENEF' sqldatatype='CHAR'>N</Attribute>
    <Attribute name='OWN_TRANS' sqldatatype='CHAR'>N</Attribute>
    <Attribute name='HOW_FIND_US' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='NO_OF_CHILD' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='DIVISION' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='NO_OF_DEPEND' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='GENDER' sqldatatype='NUMERIC'>1303874</Attribute>
    <Attribute name='LOCATION_CD' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='AVAILABILITY' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='NATIONALITY' sqldatatype='NUMERIC'>20010516</Attribute>
    <Attribute name='MARITAL_STAT' sqldatatype='NUMERIC'></Attribute>
    <Attribute name='DT_OF_BIRTH' sqldatatype='DATETIME'>09/11/1976</Attribute>
  </Property>
  <Property name='TELEPHONE' occurrence='Work'>
    <Attribute name='EXTENSION' sqldatatype='NVARCHAR'></Attribute>
    <Attribute name='TEL_NUMBER' sqldatatype='NVARCHAR'>+44 131 312 1331</Attribute>
    <Attribute name='CAN_SMS' sqldatatype='CHAR'>N</Attribute>
    <Attribute name='CAN_CONTACT' sqldatatype='CHAR'>N</Attribute>
    <Attribute name='OCC_ID' sqldatatype='NUMERIC'>2034418</Attribute>
  </Property>
```

```
</Entity>
```

To update existing entity, XML representation of the entity is used.
XML format for *Entity* update is as follows:

```
<Entity id='ENTITY_ID_TO_UPDATE' defaultrole='ENTITY_DEFAULT_ROLE'>
  <Role>Role1</Role>
  <Role>Role2</Role>
  <Role>RoleN</Role>
  <Property name='PROPERTY_NAME_MULTIPLE' bisuniqueid='BISUNIQUE_ID'>
    <Attribute name='ATTRIBUTE_NAME' sqldatatype='ATTRIBUTE_TYPE'>Value</Attribute>
  </Property>
  <Property name='PROPERTY_NAME_NAMED' occurrence='OCCURRENCE_NAME'>
    <Attribute name='ATTRIBUTE_NAME' sqldatatype='ATTRIBUTE_TYPE'>Value</Attribute>
  </Property>
  <Property name='PROPERTY_NAME_SINGLE'>
    Attribute name='ATTRIBUTE_NAME' sqldatatype='ATTRIBUTE_TYPE'>Value</Attribute>
  </Property>
</Entity>
```

Where **'BISUNIQUE_ID'** is unique ID of the *Multiple Property*
Below are the rules that apply to XML format for *Entity* update:

1. ID of the *Entity* to update must be provided. If there is no *Entity* with such in the system – **DataNotFoundException** will be thrown.
2. In order to update a record in a *Multiple Property*, **'BISUNIQUE_ID'** of the record must be specified. If there is no provided **'BISUNIQUE_ID'** in the system – **DataNotFoundException** will be thrown. If **'BISUNIQUE_ID'** is omitted – new record will be added to the property.
3. To update a N*amed Property*, '**OCCURRENCE_NAME'** or **'BISUNIQUE_ID'** must be specified. If **'OCCURRENCE_NAME'** or **'BISUNIQUE_ID'** is not found – **DataNotFoundException** will be thrown.
4. In order to delete a record from a Multiple Property, the following syntax is used:

```
<Property name='PROPERTY_NAME' bisuniqueid='BISUNIQUE_ID'>
</Property>
```

5. In order to delete a record from a Named Property, one of the following syntaxes can be used:

```
<Property name='PROPERTY_NAME' occurrence='OCCURRENCE_NAME'>
</Property>
```

- or -

```
<Property name='PROPERTY_NAME' bisuniqueid='BISUNIQUE_ID'>
</Property>
```

6. To delete all records from a property, the following syntax should be used:
   <Property name='PROPERTY_NAME_N' ></Property

Note that you can't update existent property record while deleting all records from the property – as a result property with no records will be received. E.g.:

```
<Property name='PROPERTY_NAME_N' ></Property>
<Property name='PROPERTY_NAME' occurrence='OCCURRENCE_NAME'>
  <Attribute name='ATTRIBUTE_NAME' sqldatatype='ATTRIBUTE_TYPE'>Value</Attribute>
</Property>
```

It is possible however to delete all records from the property while adding new ones.

3.4.5 getEntityData

This method retrieves existing entity data by entity ID and list of entity's properties and returns it in XML representation. The resulting XML will contain entity's data stored in the all attributes from specified properties.

```
String getEntityData( long     long_1,
                      Long     long_2,
                      String[] arrayofString_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Entity ID
arrayofString_3	String[]	Array of properties' config names to be returned

Return value: String – entity XML. See `getEntity()` for description of returned entity XML structure.

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException
- RemoteException

3.4.6 saveEntity

This method performs saving (creates new or updates existing) using given XML data.

```
String saveEntity( long   long_1,
                   String String_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Entity XML

Return value:

Type	Description
String	Entity XML

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.4.7 saveEntityData

This method performs saving an entity (creates new or updates existing) using given XML data. The return value contains only ID of the saved entity.

```
long saveEntity( long   long_1,
                 String String_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Entity XML

Return value:

Type	Description
long	Entity ID

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.4.8 deleteEntity

This method deletes existing entity. Entity can be deleted permanently if “permanently” flag is set in true.

```
void deleteEntity( long    long_1,
                   long    long_2,
                   boolean boolean_3)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Entity ID
boolean_3	Boolean	Delete Permanently (entity will be permanently deleted if set to True)

Return value: N/A

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException

3.4.9 getFavouriteEntities

This method retrieves all favourite entities of current user and returns them in XML representation. The resulting XML will contain list of entities with their Id, default role Id and default value.

```
String getFavouriteEntities(long long_1)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id

Return Value:

```
<Favourites>
  <Entity id='1' defaultrole='1'>Default value1</Entity>
  <Entity id='2' defaultrole='2'>Default value2</Entity>
  [...etc]
</Favourites>
```

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException
- RemoteException

3.4.10 getEntityDataNoPrimaryReference

This method retrieves existing entity data by entity ID and list of entity's properties and returns it in XML representation. The resulting XML will contain entity's data stored in the all attributes from specified properties. This method differs from `getEntityData()` by excluding from the returned XML attributes that represent either the occurrence ID or primary reference attribute for the property.

```
String getEntityDataNoPrimaryReference( long     long_1,
                                        long     long_2,
                                        String[] arrayOfString_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Entity ID
arrayOfString_3	String[]	Array of properties' config names to be returned

Return value: String – entity XML. See `getEntity()` for description of returned XML structure.

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException
- RemoteException

3.5 Document Interface

Document interface can be used to retrieve, create, update and delete Document in Adapt V11 database. It is also possible to get IDs of all documents for specified entity.

3.5.1 Summary of Methods

Method	Purpose
getDocument	Retrieve the details of a document including its content
saveDocument	Upload a document to the document library
deleteDocument	Delete a document from the document library
getEntityDocuments	Retrieve a list of IDs of all documents stored against an entity.
getDocumentsInfoByOwnerAndCategory	Retrieve a list of documents from a specified category stored against an entity.
getDocumentsInfoByOwner	Retrieve a list of all documents stored against an entity.

3.5.2 Objects

DocumentV1 data object

This object contains everything related to adocument in Adapt V11 system.

Field	Type	Read only	Description
documentID	Long		Document ID. Should be zero when creating new document.
owner	String	Yes	Owner name
ownerID	Long		Owner ID
category	String		Path of the document category
name	String		Document name
fileExtension	String		Document file extension
description	String		Document description
createdDate	Date		Creation date
createdBy	String	Yes	Creator name
createdByID	Long		Creator ID
updatedDate	Date		Updating date
updatedBy	String	Yes	Updater name
updatedByID	Long		Updater ID
note	String		Note
content	Byte[]		Binary content of the document
size	Date		Size of the binary content
default	Boolean		Whether document is the default one
active	Boolean	Yes	Whether document is active one(not deleted)
ownerType	Integer		Can be User = 1, Entity = 0, Journal = 2

3.5.3 getDocument

This method retrieves document from database.

```
DocumentV1 getDocument( long    long_1,
                        long    long_2,
                        boolean boolean_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Document ID
boolean_3	Boolean	Whether document content should be downloaded too

Return value:

Type	Description
DocumentV1	Document data with or without document content

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.5.4 saveDocument

This method saves document. If document ID is 0, the new document will be created.

```
long saveDocument( long long_1,
                   DocumentV1 DocumentV1_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
DocumentV1_2	DocumentV1	Document to be saved

Return value:

Type	Description
Long	ID of saved Document. Is necessary for new created documents

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.5.5 deleteDocument

This method deletes document. Document can be deleted permanently if “permanently” flag is set in true.

```
void deleteDocument( long    long_1,
                     long    long_2,
                     boolean boolean_3)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Document ID
boolean_3	Boolean	Whether the document should be deleted permanently or there should

		be a possibility to restore it

Return value: none

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException;

3.5.6 getEntityDocuments

This method returns ID of specified entity documents. Language and document library category can be specified.

```
long[] getEntityDocuments( long   long_1,
                           long   long_2,
                           String String_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Entity ID
String_3	String	Category of the document to be retrieved

Return value:

Type	Description
Long[]	Specified entity documents IDs Array

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException;

3.5.7 getDocumentsInfoByOwnerAndCategory

This method is used to retrieve list of documents for specific owner and document library category. Unlike getEntityDocuments method which returns array of document identifiers, this method returns array of document beans which contains all document information (see section 3.5.1 DocumentV1 data object) except document content.

```
DocumentV1[] getDocumentsInfoByOwnerAndCategory( long   long_1,
                                                 long   long_2,
                                                 String String_3,
                                                 String String_4 )
```

Method arguments:

Argument	Type	Description
long_1	long	Session id
long_2	long	Owner ID
String_3	String	Owner Type (‘E’ – Entity, ‘U’ – User, ‘J’ – Journal Entry, ‘T’ - Task)
String_4	String	Category of the document to be retrieved

Return value:

Type	Description
DocumentV1[]	Array of document V1 beans which contains all document information except document content

Exceptions:

- ServerErrorException
- AccessDeniedException
- DataNotFoundException
- RemoteException
- InvalidArgumentException;

3.5.8 getDocumentsInfoByOwner

This method is used to retrieve list of documents for a specific owner. Unlike getEntityDocuments method which returns array of document identifiers, this method returns array of document beans which contains all document information (see section 3.5.1 DocumentV1 data object) except document content.

```
DocumentV1[] getDocumentsInfoByOwner( long   long_1,
                                      long   long_2,
                                      String String_3 )
```

Method arguments:

Argument	Type	Description
long_1	long	Session id
long_2	long	Owner ID
String_3	String	Owner Type (‘E’ – Entity, ‘U’ – User, ‘J’ – Journal Entry, ‘T’ - Task)

Return value:

Type	Description
DocumentV1[]	Array of document V1 beans which contains all document information except document content

Exceptions:

- ServerErrorException
- AccessDeniedException
- DataNotFoundException
- RemoteException
- InvalidArgumentException;

3.6 Calendar

Calendar Interface provides access to appointments in Adapt V11 Calendar.

3.6.1 Summary of Methods

Method	Purpose
getBookingById	Retrieve details of a booking based on its ID
getBookings	Retrieve list of calendar bookings for a user or entity
saveBooking	Create/Update a booking
deleteBooking	Delete a booking
getFilteredBooking	Retrieve a list of bookings that meet a set of filter conditions

3.6.2 Objects

BookingV1 data object

This object contains everything related to calendar appointment (Booking) in Adapt V11 system.

Field	Type	Read only	Description
id	Long		Booking ID.
bookingType	Integer		USER = 0 ENTITY = 1
bookingCodeId	Long		Code ID the booking was booked with
endDate	Date		End date
startDate	Date		Start date
note	String		Booking note
subject	String		Booking subject
userIds	Long[]		User IDs
entityIds	Long[]		Entity IDs

Filter Object

Filter object to specify the filter to be used when retrieving a filtered set of bookings.

Field	Type	Required	Description
startDate	Date		Booking start date (if not specified, 01-Jan -1753 is set by default)
endDate	Date		Booking end date (if not specified, 31-Dec-9999 is set by default)
ownerID	Long	Y	Booking owner ID
ownerType	Integer	Y	Booking owner type, can be 0 - user or 1 – entity

All other fields in the Filter Object are ignored.

3.6.3 getBookingById

This method returns booking data by given booking Id.

```
BookingV1 getBookingById( long long_1,
                          long long_2,
                          int  int_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Booking ID

int_3	Integer	Booking type. Can be USER=0 or ENTITY=0

Return value:

Type	Description
BookingV1	Booking data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.6.4 getBookings

This method returns all bookings for the owner specified (entity or user).

```
BookingV1[] getBookings( long long_1,
                         long long_2,
                         int  int_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Owner ID, can be ID of Entity or User
int_3	Integer	Can be USER=1 or ENTITY=0

Return value:

Type	Description
BookingV1[]	Array of bookings

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.6.5 saveBooking

This method saves booking. If bookingID is 0, the new booking will be created.

```
long saveBooking( long      long_1,
                  BookingV1 BookingV1_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
BookingV1_2	BookingV1	Booking data to be saved

Return value:

Type	Description
long	Saved booking ID.

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.6.6 deleteBooking

This method deletes a booking.

```
void deleteBooking( long long_1,
                    long long_2,
                    int  int_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Booking ID
int_3	Integer	0 = Entity Booking or 1 = User Booking

Return value: N/A

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.6.7 getFilteredBookings

This method returns list of bookings filtered by start and end dates for the specified owner (entity or user).

```
BookingV1[] getFilteredBookings( Long   long_1,
                                 Filter Filter_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
Filter_2	Filter	Filter specifying date range and owner for which to retrieve bookings.

Return value:

Type	Description
BookingV1[]	Array of bookings

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException

ServerErrorException

3.7 Task

Task Interface provides access to Adapt V11 Tasks and is very similar to Calendar

3.7.1 Summary of Methods

Method	Purpose
getTaskById	Retrieve the details of an individual task based on its ID
getTasks	Retrieve list of tasks matching filter conditions
saveTask	Create/Update a task
deleteTask	Delete a Task
deleteTasks	Delete multiple tasks

3.7.2 Objects

TaskV1 data object

This object contains all necessary data related to task in Adapt V11 system.

Field	Type	Read only	Description
id	Long		Task ID.
creatorId	Long		ID of the entity or user who created this task
creatorType	Integer		USER = 0 ENTITY = 1
entityIDs	Long[]		ID of the entities this task is assigned to
taskCreationDate	Date		Task creation date
startDate	Date		Task start date
dueDate	Date		Task end date
completedDate	Date		Factual task completion date
completed	Boolean		Whether the task is completed
completedPercentage	Integer		Task completion percentage
status	String		Can be "All", "N", "P", "C", "H", "D"
priority	String		Can be "H", "M", "L"
subject	String		Subject of the task
description	String		Description of the task

Filter Object

Object that contains filter criteria passed to the getTasks method to sort/filter returned Tasks.

Field	Type	Required	Description
startDate	Date		Task start date
endDate	Date		Task end date
sortColumn	String	Y	Column name to sort retrieved tasks by
sortOrder	String		Order of sorting (ASC or DESC)
filterText	String		Text to search upon – applies to Task subject and documents and to Journal documents.
ownerID	Long	Y	Task owner ID
ownerType	Integer	Y	Type of the task's owner, can be 0 - user or 1 – entity

3.7.3 getTaskById

This method returns task by its ID

```
TaskV1 getTaskById ( long long_1,
```

```
                    long long_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Task ID

Return value:

Type	Description
TaskV1	Task data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.7.4 getTasks

This method returns array of task objects filtered and sorted by given criteria.

```
TaskV1[] getTasks(long   long_1,
                  Filter Filter_2,
                  int    int_3,
                  int    int_4)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
Filter_2	Filter	Filter conditions
int_3	firstItemIndex	First item Index
int_3	itemCount	Maximum number of items to be returned

Return value:

Type	Description
TaskV1 []	Array of Tasks found by given criteria

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.7.5 saveTask

This method saves a task. If taskID is 0, a new booking will be created

```
long saveTask( long   long_1,
               TaskV1 TaskV1_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
TaskV1_2	TaskV1	Task to be saved

Return value:

Type	Description
Long	Created/updated task Id

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.7.6 deleteTask

This method deletes a task.

```
void deleteTask( long long_1,
                 long long_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	ID of the task to be deleted

Return value: none
Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.7.7 deleteTasks

This method deletes tasks.

```
void deleteTask( long   long_1,
                 long[] arrayOflong_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
arrayOflong_2	Long[]	IDs of the tasks to be deleted

Return value: N/A
Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.8 Journal

Journal Interface can be used to retrieve Adapt V11 Journal entries

3.8.1 Summary of Methods

Method	Purpose
getJournalEntry	Retrieve details of a journal entry from its ID
getJournalEntries	Retrieve a list of Journal Entries that meet filter conditions

3.8.2 Objects

JournalBean Object

Object that contains all the data compromising a Journal Entry.

Field	Type	Description
boID	long	ID of the Business Object that created the journal entry
boName	String	Name of the Business Object that created the journal entry
createdDate	Date	Date the journal entry was created
creatorID	long	User ID of the user that ran the business object that created the journal entry
creatorName	String	Name of the user that created the Journal Entry
journalID	long	Journal Entry ID
note	String	Journal Entry note

Filter Object

Object that contains filter criteria passed to the getJournalEntries method to sort/filter returned Journal Entries.

Field	Type	Required	Description
startDate	Date		Return Journal Entries created on or after the start date
endDate	Date		Return Journal Entries created on or before the start date
sortColumn	String	Y	Column name to sort retrieved Journal Entries by
sortOrder	String		Order of sorting (ASC or DESC)
filterText	String		Text to search upon
ownerID	Long	Y	Journal Entry owner ID (user ID or Entity ID)
ownerType	Integer	Y	Type of the task's owner, can be 0 - user or 1 – entity

3.8.3 getJournalEntry

This method returns journal entry by ID

```
JournalBean getJournalEntry( long long_1,
                             long long_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
long_2	Long	Journal record ID

Return value:

Type	Description
JournalBean	Journal record data

Exceptions:

- AccessDeniedException
- DataNotFoundException
- ServerErrorException

3.8.4 getJournalEntries

This method returns array of journal entries found by given filter

```
JournalBean[] getJournalEntries( long   long_1,
                                 Filter Filter_2,
                                 int    int_3,
                                 int    int_4)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
Filter_2	Filter	Filter
int_3	Integer	First item Index
int_4	Integer	Maximum number of items to be returned

Return value:

Type	Description
JournalBean[]	Array of journal entries found by given filter

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9 Code Group

Used to retrieve codes and code groups

3.9.1 Summary of Methods

Method	Purpose
getCodeGroup	Retrieve information about a code group (does not include all codes in the group)
getCodeGroupByLanguage	Retrieve information about a code group. Also retrieves all codes in the group, but only for the specified language.
getCodeByID	Retrieve details of a code based on its ID
getCodeIDFromPath	Determine the ID of a code given by its path
getCodeIDFromGroupAndCodeDescr	Determine the code ID based on its group and language description
getCodeGroups	Retrieve a list of all available code groups
updateCodeGroup	Update an existing code group
updateCode	Update an existing code

3.9.2 Objects

CodeBean Object

This object contains information related to codes in Adapt V11 system.

Field	Type	Description
Id	Long	Code ID
parentId	Long	Parent code ID. 0 for the top level code
name	String	Code Name
localizedName	String	Language specific name of code
localizedDesc	String	Language specific description of code
localLanguage	String	The name of the local language used for localised settings
allDescriptions	LocalName []	An array of local name objects providing details of multilingiual name and descriptions for the code
hidden	Boolean	Is this code hidden or not
globalEquivId	Long	Language specific name of code
namespaceId	Long	Language specific description of code
synonyms	CodeSynonym[]	An array of CodeSynonym objects defining the synonyms for the code.
groups	Long[]	An array of user group IDs that visibility of this code is restricted.
dependants	DynamicVisibility[]	An array of DynamicVisibility objects defining the codes that are dependent on this code when dynamic code groups are used.

CodeGroupBean Object

This object contains information related to code group in Adapt V11 system.

Field	Type	Description
id	Long	Code group ID
groupName	String	Code group name
allDescriptions	LocalName[]	Array of language specific names represented by LocalName objects

codes	CodeBean[]	Array of codes which belong to this code group
localLanguage	String	The name of the local language used for localised settings
localisedName	String	The group name in the local language
localisedDesc	String	The group description in the local language
format	String	The format specifier for the code group
lookupBy	String	Lookup the code group based on either code or description(C – sort by code, D – sort by description)
sortBy	String	Sort the code group based on either code or description(C – sort by code, D – sort by description)
nodesExpanded	boolean	Are nodes expanded or not
allowParent	boolean	Is selection of a parent node allowed or not
namespaceID	Long	ID the namespace this code group is defined in

LocalName Object

This object contains information related to Adapt V11 system naming.

Field	Type	Description
language	String	Language name
name	String	Language specific name
description	String	Language specific description

CodeSynonym Object

The following new properties added to this object.

Field	Type	Description
synonymID	Long	The ID of the associated (synonym) code
matchPercent	Int	The extent to which it matches the code

DynamicVisibility Object

The following new properties added to this object.

Field	Type	Description
childGroupID	Long	The ID of the code group that the dependent code belongs to
childCodeID	Long	The ID of the dependent code in the group.

3.9.3 getCodeGroup

This method retrieves Code Group information identified the display name of CodeGroup with localized descriptions in all active system languages.

```
CodeGroupBean getCodeGroup( long   long_1,
                            String String_1)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id

String_1	String	Code Group name

Return value:

Type	Description
CodeGroupBean	Code Group data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9.4 getCodeGroupByLanguage

This method retrieves CodeGroup identified by domain name and display name of CodeGroup with localized representation for the language specified.

```
CodeGroupBean getCodeGroupByLanguage( long   long_1,
                                      String String_2,
                                      String String_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session id
String_2	String	Code group name
String_3	String	Code group language

Return value:

Type	Description
CodeGroupBean	Code group data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9.5 getCodeByID

This method retrieves localized representation of the Code by ID and language name

```
CodeBean getCodeByID( long   long_1,
                      long   long_2,
                      String String_3 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Code ID
String_3	String	The language to retrieve code for

Return value:

Type	Description

CodeBean	Code data

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9.6 getCodeIDFromPath

This method retrieves Code ID by Code Path

```
long getCodeIDFromPath( long   long_1,
                        String String_2 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
String_2	String	Code path. A code path consists of the code group name, followed by a '\', followed by the hierarchy within the code group with each level separated by a '\' and finally the code name. For example, "Skills\franchise\computer programmer".

Return value:

Type	Description
Long	Code ID

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9.7 getCodeIDFromGroupAndCodeDescr

This method retrieves Code ID by Code Group ID and Code Description

```
long[] getCodeIDFromPath( long   long_1,
                          long   long_2,
                          String String_3,
                          String String_4 )
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
long_2	Long	Code Group ID
String_3	String	Code description
String_4	String	The language to retrieve code for

Return value:

Type	Description
long[]	Code Ids Array

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException

3.9.8 getCodeGroups

The functionality of this method is to retrieve all the code groups available in the domain specified. CodeGroupBean object will contains all the information related to the Code Group except the codes it contains.

```
CodeGroupBean[] getCodeGroups( long long_1 )
```

Method Arguments:

Argument	Type	Description
long_1	Long	Session ID

Return value:

Argument	Description
CodeGroupBean[]	Code Group data array

Exceptions:

- AccessDeniedException
- ServerErrorException
- DataNotFoundException

Below is an example of XML request that retrieves an array of CodeGroupBean objects.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cod="http://codegroup.webservice.bis.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <cod:getCodeGroups>
         <String_1>SRxUKCRA</String_1>
      </cod:getCodeGroups>
   </soapenv:Body>
</soapenv:Envelope>
```

3.9.9 updateCodeGroup

The functionality of this method is to update the code group details in the form of CodeGroupBean object passed as parameter to this method along with the domain name. The CodeGroupBean contains all the data related to the code group except the codes.

```
long updateCodeGroup( long          long_1,
                      CodeGroupBean CodeGroupBean_2)
```

Method arguments:

Argument	Type	Description
long_1	Long	Session ID
CodeGroupBean_2	CodeGroupBean	Code Group details

Return value:

Type	Description
long	Id of the code group

Exceptions:

- InvalidArgumentException
- AccessDeniedException
- DataNotFoundException
- ServerErrorException
- InvalidCodeGroupException

Below is an example of a SOAP request that updates the passed codegroupbean details.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cod="http://codegroup.webservice.bis.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <cod:updateCodeGroup>
         <String_1>SRxUKCRA</String_1>
         <CodeGroupBean_2>
            <allDescriptions>
               <description>Authorisation Italian Desc</description>
               <language>Italian</language>
               <name>Authorisation Italian</name>
            </allDescriptions>
            <allowParent>true</allowParent>
            <format>YYYY</format>
            <groupName>Authorisation</groupName>
            <id>7959152</id>
            <localLanguage>English</localLanguage>
            <localisedDesc>Authorisation English Desc</localisedDesc>
            <localisedName>Authorisation English</localisedName>
            <lookupBy>D</lookupBy>
            <namespaceID>0</namespaceID>
            <nodesExpanded>true</nodesExpanded>
            <sortBy>D</sortBy>
         </CodeGroupBean_2>
      </cod:updateCodeGroup>
   </soapenv:Body>
</soapenv:Envelope>
```

3.9.10 updateCode

The functionality of this method is to update the language specific code details in the form of CodeBean object passed as parameter to this method along with the domain name. The CodeBean object contains all the data related to the code.

```
long updateCode( long     long_1,
                 CodeBean CodeBean_2 )
```

Method arguments:

Argument	Type	Description

long_1	Long	Session ID
CodeBean_2	CodeBean	Code details

Return value:

Type	Description
long	Id of the code

Exceptions:

- InvalidArgumentException
- AccessDeniedException
- DataNotFoundException
- ServerErrorException
- InvalidCodeException

Below is an example of a SOAP request that updates the language specific codebean details. The language name is specified in between 'localLanguage' tag is highlighted.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cod="http://codegroup.webservice.bis.com/">
   <soapenv:Header/>
   <soapenv:Body>
      <cod:updateCode>
         <String_1>SRxUKCRA</String_1>
         <CodeBean_2>
            <allDescriptions>
               <description>CV Spanish Desc</description>
               <language>Spanish</language>
               <name>CV Spanish</name>
            </allDescriptions>
            <dependants>
               <childCodeID>4611686018435640346</childCodeID>
               <childGroupID>9187343239844111142</childGroupID>
            </dependants>
            <globalEquivId>0</globalEquivId>
            <groups>6624762</groups>
            <hidden>true</hidden>
            <id>8249277</id>
            <localLanguage>Italian</localLanguage>
            <localizedDesc>CV Italian Desc</localizedDesc>
            <localizedName>CV Italian</localizedName>
            <name>CV_TYPE</name>
            <namespaceId>0</namespaceId>
            <parentId>0</parentId>
            <synonyms>
               <matchPercent>55</matchPercent>
               <synonymID>8249602</synonymID>
            </synonyms>
         </CodeBean_2>
      </cod:updateCode>
   </soapenv:Body>
</soapenv:Envelope>
```

3.10 Meta Data – IMetaDataServiceV1

MetaData service interface can be used to retrieve meta data information from the system.

3.10.1 Summary of Methods

Method	Purpose
getRoles	Retrieve a list of all available Roles

3.10.2 Objects

RoleBean Object

This object contains information related to roles in Adapt V11 system.

Field	Type	Description
id	long	Role ID
name	String	Configuration role name.
roleGroupId	long	Role Group ID (if available)
roleGroupName	String	Role Group confuguration name
descriptionsArray	LanguageDescription[]	An array of role's multilingual descriptions

LanguageDescription

This object contains information related to multilingual names & descriptions in Adapt V11 system.

Field	Type	Description
languageId	long	Language ID
LanguageName	String	Language name
name	String	Item's name
description	String	Item's description

3.10.3 getRoles

The purpose of this method is to retrieve all the roles available in the specified domain. RoleBean object will contain all the information related to the Role, including role's ID and config name, parent group ID and config name, and array of role's multilingual descriptions, wrapped into the LanguageDescription objects.

```
RoleBean[] getRoles(long long_1)
```

Method arguments:

Argument	Type	Description
long_1	long	Session id

Return value:

Type	Description
RoleBean[]	Roles data array

Exceptions:

- InvalidArgumentException
- AccessDeniedException
- DataNotFoundException

- ServerErrorException
- InvalidCodeException

3.10.1 RoleBean

3.11 DataAdmin ManageRequestNotification

Is used to create merge request and create delete request

3.11.1 Summary of Methods

3.11.2 Objects

ManageRequestBean Object

3.11.3 createMergeRequest

The functionality of this method is to create the merge request and it contains merge request details in the form of ManageRequestBean object passed as parameter to this method along with the session id.

```
String createMergeRequest( long              long_1,
                           ManageRequestBean ManageRequestBean_2 )
```

Method arguments:

Argument	Type	Description
long_1	long	Session id
ManageRequestBean_2	ManageRequestBean	Merge request details to be saved

Return value:

Type	Description
String	Success/failure create merge request message

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException
- RemoteException

3.11.4 createDeleteRequest

The functionality of this method is to create the delete request and it contains delete request details in the form of DeleteRequestBean object passed as parameter to this method along with the session id.

```
String createDeleteRequest( long              long_1,
                            DeleteRequestBean DeleteRequestBean_2)
```

Method arguments:

Argument	Type	Description
long_1	long	Session id
DeleteRequestBean_2	DeleteRequestBean	delete request details to be saved

Return value:

Type	Description
String	Success/failure create delete request message

Exceptions:

- AccessDeniedException
- InvalidArgumentException
- DataNotFoundException
- ServerErrorException
- RemoteException