CA Business Service Insight

Software Development Kit (SDK) Reference Guide

8.2.5

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

This section contains the following topics:

Target Audience

The target audience are developers and programmers who wish to use the CA Business Service Insight Software Development Kit (SDK) in integrating CA Business Service Insight functionality with enterprise portals or other applications.

In This SDK

This SDK provides the tools required to integrate CA Business Service Insight into your portal or any other system such as LDAP, CMDB, and so on. This SDK is for use with CA Business Service Insight and includes:

- Single Sign-on (see page 11) provides useful information for implementing single sign-on.
- Authentication Gateway (see page 15) provides basic authentication and password management capabilities.
- LDAP Integration (see page 21) provides integration steps and methods for implementing <g> in LDAP systems.
- CMDB Integration (see page 23) provides CMDB integration steps and methods for implementing resource management in <g>.
- Integration (see page 29) provides the API required for implementing translation scripts.
- Portal Integration (see page 133) provides an overview of the portal integration data flow, the API and methods required for SOA implementations, and the CA Business Service Insight web parts that can be used for integration with Microsoft SharePoint.
- Open API (see page 179) provides connectivity with CA Business Service Insight, using standard secured web services.

Chapter 2: Single Sign-On (SSO)

Single sign-on (SSO) is a mechanism in which a single action of user authentication and authorization permits a user to access all systems to which he/she has access permission, without the need to enter multiple passwords.

Once a user logs on to the portal using SSO, CA Business Service Insight users can directly access CA Business Service Insight through the portal.

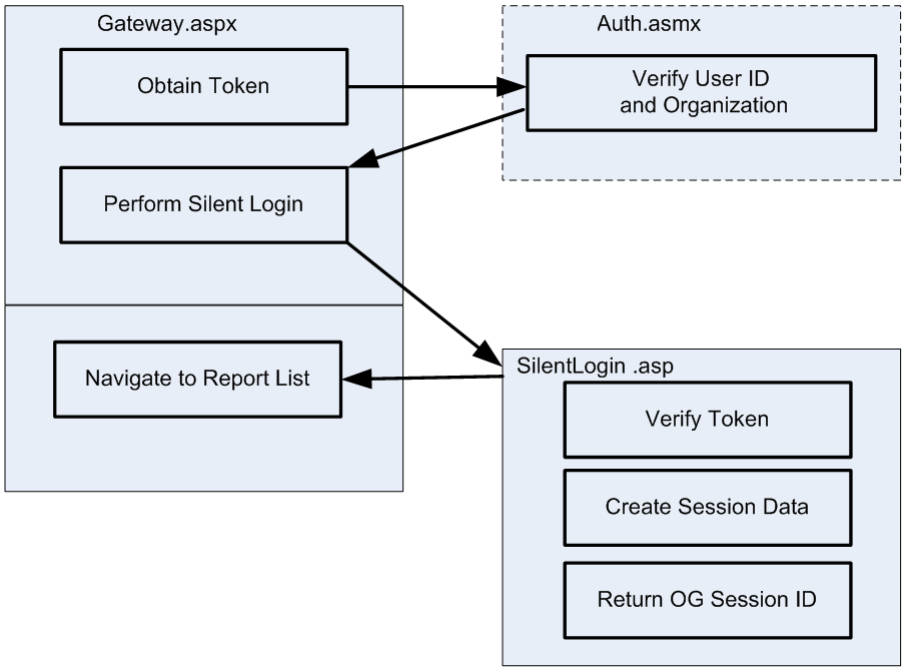
This chapter provides guidelines for implementing SSO, including information regarding:

- Secured silent sign-on (allowing SSO to be invisible to the end user)
- Authentication XML Web service (supporting the secure SSO page)

This section contains the following topics:

SSO Event Flow

The following image illustrates the internal event flow in the SSO integration process.



Once a user signs-on to the portal, SSO checks the user credentials against the CA Business Service Insight database, ensuring secure access to Insight. This is done using the Gateway.aspx, which resides on the portal Web server, in the CA Insight Portal Client virtual directory.

Gateway.aspx:

- Retrieves the current portal user name and organization (in terms of CA Insight) using the portal API. For example, SAP Portal API or Active Directory.
- Calls the http://Insight/WebServices/Auth.asmx Web service, passing the user name and organization as input parameters.

The Insight/Web Services directory is configured on the CA Insight Web server as accessible only from portal IPs using IP Address and Domain Name Restrictions on IIS.

Auth.asmx checks if a user with such credentials (user name and organization) exists and is active in the CA Insight database. If such a user exists, a special encrypted token containing the user ID is returned and a timestamp is issued. If no such active user exists, a null token is returned.

SilentLogin.asp (part of the CA Business Service Insight installation) verifies the token and navigates to the report list (or other portal content), creates session data and context, and returns an CA Business Service Insight Session ID. If the token is obsolete or incorrect, it returns an error message.

To implement SSO, Insight inherits the user name and organization from an organizational Identity Management Solution (IMS) such as the Microsoft Active Directory. The IMS must be available and accessible to receive and respond to Insight requests for authentication.

Secured Silent Logon

Secured silent sign-on allows the SSO to be invisible to the end user.

SSO Integration Steps

A few transactions should be configured and applied to connect the CA Business Service Insight login system to the IMS. These SSO integration steps include:

1. When the user is logged into CA Business Service Insight (via portal or any environment containing user credentials), CA Business Service Insight obtains and extracts the user ID and organization from the token or ticket.

For example, when the user is logged into the portal, an appropriate token is attached to the user session. The token contains the user information that can be used by CA Business Service Insight. Configure this step in the InsightGateway.aspx page. The coding in the InsightGateway.aspx page and the implementation method depend on the Identity Management tokens.

The following sample Gateway.aspx.cs shows an example of the GetInsightUserCredentials function:

```
/// <summary>
/// Obtain CA Business Service Insight user name and organization from portal
user directory
/// This method is supposed to call ActiveDirectory or another repository using
portal API
/// to obtain current user name and organization in terms of CA Business Service
Insight
/// </summary>
/// <returns>CA Business Service Insight user credentials
struct</returns>
private UserCredentials GetInsightUserCredentials ()
{
UserCredentials ucInsightUser = new
UserCredentials () ;
//currently always assume user is sadmin and organization is Insight (default)
ucInsightUser.UserName = "Insightuser";
ucInsightUser.Organization = "Acme";
return ucInsightUser;
}
```

2. Once the user information is retrieved from the token, the user is authenticated with the CA Business Service Insight security repository system using InsightAuth.asmx page. This page receives the user ID and the organization from the token extraction in the previous step.

3. This step is performed in InsightSilentLogin.asp page. After getting CA Business Service Insight approval for the user ID, CA Business Service Insight creates a secure session ID and context objects, which follows the user when browsing. The session ID object contains security and verification information. In addition, the system creates Context object, including locale and other regional setting.

Chapter 3: Authentication Gateway

This section contains the following topics:

Overview

CA Business Service Insight provides basic authentication and password management capabilities.

There are two options to enhance CA Business Service Insight's authentication capabilities.

- Using the single sign-on, as described in Single Sign-On (SSO) (see page 11). In this option, the consideration is that the authentication and password management is performed using an external system and when logging into CA Business Service Insight, no authentication is required.
- Developing a set of web services that enhances the authentication capabilities (as described in this chapter) following a known interface. Simply configured, these web services are called automatically on every login trial into CA Business Service Insight and on every password change. Such an implementation may, for example, enable restrictions as follows:
  - Password cannot be identical to the last 12 passwords.
  - Password must be at least six characters long and must contain both alphabetic and non-alphabetic characters.
  - After four login failures the user is blocked.

CA Business Service Insight provides the following three services:

- **Login Gateway**

  Web service that should be developed. It is called prior to CA Business Service Insight's standard authentication.

- **Password Change Gateway**

  Web service that should be developed. It is called upon password change by either the administrator or the user.

- **Deactivate User Method**

  Web service used to deactivate a user in CA Business Service Insight. It blocks the user from logging into the system. The administrator can reactivate the user using the standard user interface.

Web Services Interface

CA Business Service Insight provides two interfaces that need to be implemented in order to extend the authentication restrictions:

- Authenticate - is called before the standard CA Business Service Insight authentication.
- EnforcePasswordPolicy - is called upon any password change.

The following is a detailed description of the interfaces that need to be implemented by the developer, noting the data structure member order.

```
[WebService(Namespace = "http://Insight.com")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public interface IAuthenticationService
{
    [WebMethod]
    AuthenticationResult Authenticate(UserCredentials user);

    [WebMethod]
    AuthenticationResult EnforcePasswordPolicy (UserCredentials user);
}
public sealed class UserCredentials
{
    public string Username;
    public string Password;
    public string Organization;
}

public sealed class AuthenticationResult
{
    public bool IsAuthenticated;
    public string ErrorMessage;
}
```

The following is an example for implementing the above web services:

```
using System;
using System.Web;
using System.Collections;
using System.Web.Services;
using System.Web.Services.Protocols;



/// <summary>
/// Summary description for PreAuthenticationExample
/// </summary>
[WebService(Namespace = "http://CA.com")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public class PreAuthenticationExample : System.Web.Services.WebService
{
    public class UserCredentials
    {
        public string Username;
        public string Password;
        public string Organization;
    }

    public sealed class AuthorizationResult
    {
        public bool IsAuthenticated;
        public string ErrorMessage;
    }

    public PreAuthenticationExample()
    {

        //Uncomment the following line if using designed components
        //InitializeComponent();
    }

    [WebMethod]
    public AuthorizationResult Authenticate(UserCredentials user, string state)
    {
        AuthorizationResult Result = new AuthorizationResult();

        if (user.Password == "DeactivateMe")
        {
            Result.IsAuthenticated = false;
            Result.ErrorMessage = "The user became inactive. Contract your system
administrator";


            UserManagementService Um = new UserManagementService();
```

```
            Um.DeactivateUser(user.Username, user.Organization);
        }
        else
        {
            Result.IsAuthenticated = true;
        }

        return Result;
    }

    [WebMethod]
    public AuthorizationResult EnforcePasswordPolicy(UserCredentials user, string
state)
    {
        AuthorizationResult Result = new AuthorizationResult();
        //Do not allow password that is identical to username or shorter than 7
characters
        if (user.Username==user.Password || user.Password.Length<7)
        {
            Result.IsAuthenticated = false;
            Result.ErrorMessage = "Invalid password.";
        }
        else
        {
            Result.IsAuthenticated = true;
        }

        return Result;
    }
}
```

Implementing the Authentication Gateway

To implement the authentication gateway:

1. Develop a web service as indicated in Web Services Interface (see page 16).
2. In CA Business Service Insight, go to **Administration > Preferences > Advanced > System > Security** and set the value of the *Enable Pre Authentication* parameter to **Y**.
3. In CA Business Service Insight, go to **Administration > Preferences > Advanced > System > Security** and set the value of the *Pre Authentication Web Service URL* parameter to the path where the web service file created in step 2 is located.

Deactivating a User

Deactivating a user is enabled using the *DeactivateUser* method. This method, located in **/WebServices/UserManagementService.asmx**, is utilized to block a user from logging into CA Business Service Insight. For example, users performing four consecutive failed logins are blocked.

```
void DeactivateUser(string username, string organization)
```

This method receives two arguments, *user name* and *organization*. It does not return a value (it is *void*).

The web service namespace is: **http://CA.com**.

Notes:

- Securing this web service is a matter of implementation.
- Reactivating a user is performed by the administrator through CA Business Service Insight's standard user interface.

Chapter 4: LDAP Integration

This section contains the following topics:

Overview

CA Business Service Insight keeps the relevant security information related to the service delivery process in addition to the user name and organization. The association with the relevant contract parties/customers as well as the set of permitted activities in the service delivery process (defined by the role definition) should be kept in CA Business Service Insight in addition to the existing LDAP system.

To minimize the manual activities involved in the creation and management of users in CA Business Service Insight, an integration and synchronization process should be defined between CA Business Service Insight and the LDAP system. CA Business Service Insight provides an open interface for the implementation of the integration process.

LDAP Integration Steps

To integrate the LDAP system with CA Business Service Insight, the following steps should be performed:

1. Identify the appropriate organizational LDAP system (for organizations that have more than one LDAP system).
2. Analyze how CA Business Service Insight users are defined and grouped in the LDAP system in order to gather the information using SQL queries.
3. Use CA Business Service Insight translation scripts and configure the appropriate scripts to create users, user groups and their reference to contract parties and roles.
4. Ensure that the translation script supports additions, modifications, and deletions of users and user groups.
5. Schedule the translation scripts to run periodically.

Note: Alternatively, if there are no LDAP systems available, users can be managed manually in CA Business Service Insight as a replacement for the LDAP integration process.

LDAP Integration Methods

The available LDAP integration methods are listed in the following four tables.

Organization Methods	
AddOrganization (see page 70)	IsOrganizationExists (see page 71)

Role Methods	
IsRoleExists (see page 113)	SearchRoles (see page 114)

User Methods	
AddUserByMap (see page 119)	GetUserName (see page 123)
GetOrganizationName (see page 70)	IsUserExists (see page 124)
GetUserDetails (see page 122)	SearchUsers (see page 124)
GetUserFullName (see page 123)	UpdateUserByMap (see page 125)

User Group Methods	
AddUserGroupByMap (see page 126)	IsUserGroupExists (see page 128)
DeleteUserGroup (see page 127)	SearchUserGroups (see page 129)
GetUserGroupDetails (see page 127)	UpdateUserGroupByMap (see page 129)

See Translation Scripts-User Function Scripts Example (see page 325) for a user function script example.

Chapter 5: CMDB Integration

This section contains the following topics:

Overview

CA Business Service Insight Service Delivery CMDB (also known as Resource Management) contains information about the underlying components that make up the physical and logical infrastructure used to deliver the services. The main purpose of the Service Delivery CMDB is to allocate resources and related them to a service and contract party, to group resources into logical groups, to define attributes for resources, and to capture all these changes over time.

In addition to manual change controls, the Service Delivery CMDB provides an automated solution to synchronize the changes from external CMDBs (one or many). The automation process links and reflects the external CMDB changes into the Service Delivery CMDB. Advanced users can define conditional business logic to automatically change resource attributes, link resources to contract parties and services and even create and manage the contract parties and services themselves.

It is important to distinguish between the CMDB integration process and the translation process. The translation process aims to make the translation between the underlying physical and logical components identified by the adapters during the performance data gathering process into CA Business Service Insight entities such as resources and event types. The input for this process is translation entries, which are created by the running adapters. This translation is made either manually with the intervention of an interactive user, or automatically by using conditional business logic. In some cases the translation process acts as the active integration process to create and manage resources in CA Business Service Insight. This may happen for two reasons: either there is no CMDB in place or there is no CMDB integration process in place.

Both the integration process and the translation process use the integration framework and open interface.

CMDB Integration Steps

To integrate the CMDB system with CA Business Service Insight, the following steps should be performed:

1. Identify the appropriate organizational CMDB system (for organizations that have more than one CMDB system).
2. Analyze how the relevant resources (configuration items) are defined and grouped in the CMDB system in order to gather the information using SQL queries.
3. Use CA Business Service Insight translation scripts and configure the appropriate scripts to create resources and resource groups and their association with contract parties and services. If necessary, configure the script to create contract parties and services.
4. Ensure that the translation script support additions, modifications, and deletions of resources, resource groups, contract parties, and services.
5. In case of additions, modifications, and deletions related to resources and resource groups, ensure that they are all applicable to a well-known effective date or to a predefined change set.
6. Schedule the translation scripts to run periodically.

Automatic Translation Process Steps

To automate the translation process in CA Business Service Insight, the following steps should be performed:

1. Identify the relevant translation table.
2. Use CA Business Service Insight translation scripts and configure the appropriate scripts to run over the pending translation entries to be translated to the relevant entities in CA Business Service Insight based on name or based on some business logic.
3. In case the translation script should create the relevant entities, gather the relevant information from either the translation entries themselves or access to an external system using SQL queries.
4. Schedule the translation scripts to run periodically.

CMDB Integration Methods

The available CMDB integration methods are listed in the following 11 tables.

Change Set Methods	
AddChangeSet (see page 49)	UpdateChangeSet (see page 51)

CommitChangeSets (see page 61)	AddResourceVersion (see page 110)
GetChangeSetDetails (see page 49)	IsResourceVersionExists (see page 111)
IsChangeSetExists (see page 50)	IsPreliminaryResourceVersion (see page 111)
SearchChangeSets (see page 51)	SearchResourceVersions (see page 112)

Contract Party and Contract Party Group Methods

AddContractPartyByMap (see page 52)	IsContractPartyGroupExists (see page 57)
AddContractPartyGroupByMap (see page 56)	SearchContractParties
GetContractPartyDetails (see page 53)	SearchContractPartyGroups (see page 58)
GetContractPartyGroupDetails (see page 56)	UpdateContractPartyByMap (see page 55)
IsContractPartyExists (see page 54)	UpdateContractPartyGroupByMap (see page 58)

Domain Category Methods

GetDomainCategoryDetails (see page 59)	SearchDomainCategories (see page 60)
IsDomainCategoryExists (see page 59)	

Event Type Methods

AddEventTypeByMap (see page 61)	IsEventTypeExists (see page 63)
GetEventTypeDetails (see page 62)	

Exception Methods

AddException (see page 64)	SearchExceptions
DeactivateException (see page 66)	

General Methods

Commit (see page 67)	Log (see page 68)
CreateMap (see page 67)	Rollback (see page 69)
IsTimeZoneExists (see page 68)	

Resource Methods

AddResource (see page 72)	GetResourceDetailsByDate (see page 77)
AddResourceByMap (see page 72)	GetResourceGeneralDetails (see page 78)

AttachResourcesToContractParties (see page 42)	IsResourceExists (see page 82)
AttachResourcesToServices (see page 44)	IsResourceExistsInResourceVersion (see page 83)
CommitResources (see page 75)	IsResourceExistsInTime (see page 82)
DetachResourcesFromAllContractParties (see page 44)	SearchResources (see page 83)
DetachResourcesFromAllServices (see page 45)	UpdateResourceGeneralDetails (see page 88)
DetachResourcesFromContractParties (see page 46)	UpdateResourcesCustomAttribute (see page 84)
DetachResourcesFromServices (see page 48)	UpdateResourcesEffective (see page 85)
GetResourceDetails (see page 75)	

Resource Group Methods

AddResourceGroup (see page 99)	GetResourceGroupGeneralDetails (see page 103)
AddResourceGroupByMap (see page 100)	IsResourceGroupExists (see page 104)
AttachResourcesToResourceGroups (see page 42)	IsResourceGroupExistsInResourceVersion (see page 104)
DetachResourcesFromAllResourceGroups (see page 45)	IsResourceGroupExistsInTime (see page 103)
DetachResourcesFromResourceGroups (see page 47)	SearchResourceGroups (see page 105)
GetResourceGroupDetails (see page 100)	UpdateResourceGroupGeneralDetails (see page 108)
GetResourceGroupDetailsByDate (see page 102)	

Resource Type Methods

AddResourceType (see page 108)	GetResourceTypeDetails (see page 109)
AttachResourcesToResourceTypes (see page 43)	IsResourceTypeExists (see page 110)
DetachResourcesFromResourceTypes (see page 47)	

Service Methods	
AddService (see page 114)	IsServiceExists (see page 116)
GetServiceDetails (see page 115)	SearchServices (see page 116)

Translation Entry Methods	
DeleteEntry (see page 117)	SetManualTranslationEntry (see page 118)
IgnoreEntry (see page 117)	TranslationEntry (see page 119)
RestorePendingEntry (see page 118)	

See CMDB Example (see page 330) for a script example.

Chapter 6: Integration

This section contains the following topics:

Overview

CA Business Service Insight provides an open interface for several domains in the system, mainly for resource management and the security module. This open interface enables integrations with CMDB and LDAP systems. For more information, see LDAP Integration (see page 21) and CMDB Integration (see page 23).

In addition to the open interface CA Business Service Insight provides a development framework, called Translation Scripts, for the purpose of integration scripts development.

The open interface and the development framework also support the automation of the translation process, in which components are identified by the adapters and sent in a form of translation entries, mapped and translated, into CA Business Service Insight entities such as resources and event types.

Automation of the translation process and the integration decreases time consumed in both system initialization process and on-going activities. The quality of the content entered into the system is increased as the human errors resulting from intensive manual work are prevented.

It is important to state that the development of integration scripts requires advanced skills and as such increase the complexity of the implementation.

This chapter provides the following:

- Description of the Integration Development Framework.
- An API guide that details all methods supported by the open interface.

Integration Development Framework

The development framework provides several tools to facilitate the process of integration scripts development:

- **Script Editor**

  CA Business Service Insight integrated editor from development of scripts. For more information, refer to the current CA Business Service Insight *User Guide*.

- **Translation Entries Filter**

  User interface for defining the filter of the translation entries on which the script will work. The following entry filtering criteria are available:

  - Specific translation table
  - Translation entry status (pending/all)
  - Additional filtering on any of the entry fields

- **Script Scope**

  Built-in tool to test the developed script. When testing the script in the script scope, no operation used by the formal open interface is made in the database. For more information, refer to the current CA Business Service Insight *User Guide*.

- **Script Execution**

  Using the user interface, the user can either run the script or schedule it. The following alternatives are available:

  - Define the script to be executed when receiving a new Translation request (a new translation entry).
  - Define the script to run every X hours/days.
  - Run the script from the scripts page (run now option).
  - Run the script on selected translation entries in the Translation entries page.

  In the first three cases above, the script runs only on entries defined in the filter. In the last case, the script runs only on the selected entries.

  The user can define the order between the various scripts when happen to run in the same time.

- **Script Runtime Environment**

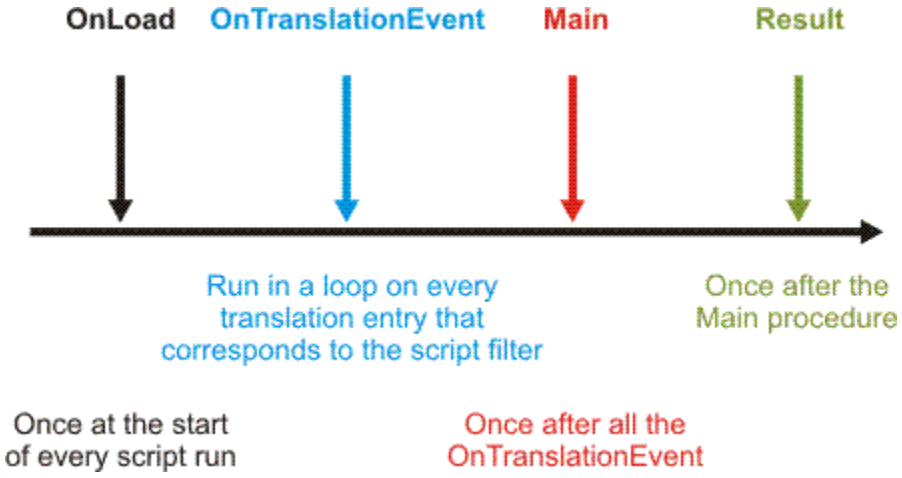
  The script is run under a run-time environment that provides a shell to the developed script. In addition, a library of methods and objects assisting the script author when creating the script is provided out-of-the-box.

  The script run time contains several procedures and functions that act as event handlers triggered to be executed on a fixed order. These event handlers are optional and the script author may use only a portion of them.

  Procedures and functions include:

- OnLoad Procedure
- OnTranslationEvent (EntryDetails) Procedure
- Main Procedure
- Result function

Note: All dates in the script use the same time zone and date format, according to the defined script time zone. The time zone is defined in the general details of the script.



Name	Description
OnLoad	Called once when the script starts. It can be used for initialization of global variables.
OnTranslationEvent	Called for every translation entry that answers to the script filter. It gets one parameter as input parameter: EntryDetails. (This parameter is a TranslationEntry component).
Main	Called once after the OnLoad and after the OnTranslationEvent functions. It can be used to write code that reads data from external data source and updates the resources, event types, services, and contract parties tables in CA Business Service Insight.
Result()	Function that is called after the Main procedure. It returns a string that will be inserted to the LastResult field that can be viewed in the translation scripts list page or in the translation script details page.

In addition to the above, the following objects are provided as part of the framework library:

- **Map**

  A class of objects that can be created using the Tools.CreateMap method. This class of objects represents a unique, sorted container that associates a string type key with a value of any type. No two elements in a class Map object may have the same key.

- **SafeODBC**

  This safe component lets the user connect to any database using ODBC for reading purposes only.

- **EntryDetails**

  This component represents the translation entry fields. It supplies a Get method for each field in the entry. The TranslationEntry component represents one translation entry. The component supplies only Get methods. The OnTranslationEntry method in the script gets one parameter called EntryDetails, which is a TranslationEntry component.

- **Tools**

  The object contains all methods that are part of the CA Business Service Insight open interface.

  There are two methods managing database transactions that are part of the tools object and require special attention. These methods are Commit and Rollback. The runtime environment manages the opening of a database transaction. Inside the script the user can manage transaction directly. This is to assure that blocks of data are inserted successfully as complete units. The user can either commit or roll back the changes at any point. In case a commit operation is not performed due to an error or if it was not called explicitly, the script is rolled back automatically. It's important to mention that testing the script in the script scope the commit operation is not performed in any circumstances.

For more information, see API Reference List (see page 33).

Safe and unsafe running modes

A script that is indicated as Safe **cannot** do any harm to the environment where run, by accessing directly the database or the file system. In addition, it cannot use CA Business Service Insight internal components (CA Business Service Insight COM objects). Safe script can use only safe objects that were defined by CA Business Service Insight in advance. The safe mode is the default mode.

A script can be changed to run in unsafe mode, and can then access databases in write-mode, create and delete files from the file system, and use CA Business Service Insight internal components (CA Business Service Insight COM objects). A script can be changed to unsafe mode only directly in the database, so as to prevent unauthorized users from creating such scripts. See also: LDAP Integration Steps (see page 21) and Translation Scripts-User Function Scripts Examples (see page 325).

Important Note: CA Business Service Insight COM objects have an XML interface that can change between versions. When using the methods of CA Business Service Insight COM objects, these are handled by their transactions and open other transactions. It is in these subsequent transactions that the commit will be done whenever a method is finished. Updates already committed will not be rolled back if an error occurs in the middle of the script, only the updates in the effected method will be rolled back. Furthermore, the XML interface of the CA Business Service Insight COM object may change and will cause the script to produce run time errors. The use of the unsafe mode is recommended in scripts only for special cases, such as reading from external data sources.

API Reference List

Translation Script Objects

By default, the translation script runs in safe mode using the safe objects listed in this section.

Translation Script Overview

Insights into existing commit rules & behavior

The following insights may help understanding the existing resource/ changeset commit behavior.

1. A resource/resource group cannot have multiple uncommitted changes in two separate change sets.
2. A resource/resource group cannot have multiple uncommitted changes over two separate dates.
3. Changing any of the following fields of a resource/resource group (from a particular effective date) cause the resource (and only that particular resource) to have uncommitted changes:
   - Effective
   - Resource Type
   - Service Components

- Contract Parties
- Any Custom Attributes

4. Changing any of the following fields of a resource/resource group (from a particular effective date) cause the resource and the related resource/resource groups to have uncommitted changes:
   - Resource Groups
   - Resource Members
5. When creating a new resource/resource group, a change set is passed. Two pieces of information are taken from this change set:
   a. The change set that the uncommitted change should be in, and
   b. The effective date from the resource is copied from the change set at the time of creating a new resource

   - Due to this point (b), a change set's effective date can be changed multiple times before the change set is committed.
   - We recommend calling *Tools.CommitChangeSet* or *Tools.CommitResources* after every 500-1000 changes, in order to keep the duration and size of the Tools.CommitChangeSet or Tools.CommitResources to a minimum.
   - Because *Tools.CommitChangeSet* (or *Tools.CommitResources*) impacts performance, we recommend not calling it after every change, but instead, after the last change.
   - It is possible to execute multiple (e.g., 500) of the following actions one after another, without requiring an intermediate commit change set or resources, since only the current resource being added/updated is effected, even if each of the actions refer to a different effective dates (assuming the same resource is not repeated).
   - Add a Resource (which is not within a Resource Group).
   - Add a Resource Group (which is not within a Resource Group, and which does not have Resource members).
   - Change resource effectiveness (ie Effective = Yes/No).
   - Change associated resource types / service components / contract parties.
   - Change resource custom attribute value(s).

Other miscellaneous resource rules

- A resource must be attached to at least one resource type.
- A resource cannot have a recursive relationship to itself, even though multiple resource groups.
- Updating resource general details is independent of change set and uncommitted changes.

Use of "Tools.CommitChangeSets" or "Tools.CommitResources"

Ideally "Tools.CommitChangeSets" or "Tools.CommitResources" should be called only after the change set has over 500 uncommitted resources, or at the end of the script.

Reasons to call "Tools.CommitChangeSets" other than this are before/after are:

- A resource is updated for the second time
- A call to Tools.AttachResourcesToResourceGroups
- A call to Tools.DetachResourcesToResourceGroups
- A call to Tools.AddResourceByMap - where the map includes non null value(s) for "ResourceGroups"
- A call to Tools.AddResourceGroupByMap - where the map includes non null value(s) for "ResourceGroups" or "Resources"
- A call to Tools.UpdateResourceByMap - where the map includes non null value(s) for "ResourceGroups"
- A call to Tools.UpdateResourceGroupByMap - where the map includes non null value(s) for "ResourceGroups" or "Resources"
- A call to Tools.AddResource - with non null / non empty ResourceGroups
- A call to Tools.AddResourceGroup - with non null / non empty Resources

Map

A Map is a class of objects that can be created using the Tools.CreateMap method. This class of objects represents a unique, sorted container that associates a string type key with a value of any type. No two elements in a class Map object may have the same key.

(<Key>)=<Value>

Inserts an element including a Key and a Value into the Map. If the Key already exists, the old Value is replaced with the new one.

(<Key>)

Returns a value associated with a Key. If the key is not found, empty is returned.

Item(<Key>)=<Value>

Inserts an element with a key and a value into the map. If the key already exists, the old value is replaced with the new one.

Item(<Key>)

Returns a value associated with a key. If the key is not found, empty is returned.

Count

Returns the number of objects currently stored in the map.

Empty

Returns True if the number of objects in the Map is zero; otherwise, returns False.

Erase(<Key>)

Removes the value associated with the key from the map.

Clear

Removes all elements from the map.

Exist(<Key>)

Returns True if an element with the Key exists; otherwise, returns False.

Dump

Returns a string containing all map information in a readable format.

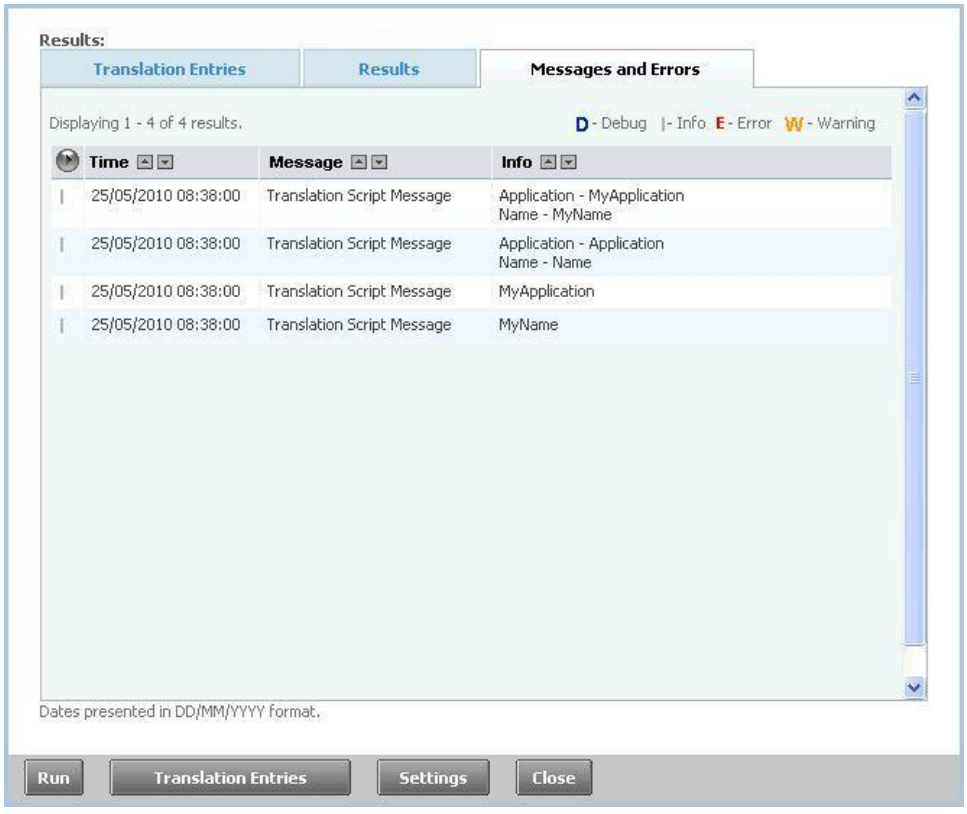
KeysSet

Used after a Map has been populated with keys, each with its own values.

The KeySet method can assist in iterating through the keys by creating a new Map whose keys and values are equal to the keys of the original map.

Example:

```
Dim myMap
    Set myMap = Tools.CreateMap
    myMap("Name") = "MyName"
    myMap("Application") = "MyApplication"
    Tools.Log myMap.dump , "I"
    Dim keySet
    Set keySet = myMap.KeysSet
    Tools.Log keySet.dump , "I"
    Dim element
    For Each element In keySet
          Tools.Log myMap(element), "I"
    Next
```



EntryDetails

This component represents the translation entry fields. It supplies a Get method for each field in the entry.

The TranslationEntry component represents one translation entry. The component supplies only Get methods. The OnTranslationEntry method in the script gets one parameter called EntryDetails, which is a TranslationEntry component.

Each of the following methods (or attributes) returns the value of a specific field in the translation entry, except for AllFieldsValues, Dump, and NumberOfFields.

Attribute	Description
EntryId	Translation entry ID.
TableId	Translation table ID.
TableName	Translation table name.
TableType	Translation table type. Must always use capital letters (legal values).
TableReturnType	The Return type. Must always use capital letters (legal values).
Status	Must always use capital letters (legal values). One of the following values: {"PENDING","TRANSLATED","IGNORED","DELETED"}
IsManualTranslation	Manual translation (legal values).
NumberOfFields	The number of input fields.
FieldName	Gets one parameter: field number.
FieldValue	Gets one parameter: field number.
ResourceId	Resource ID (only if it translated to, otherwise id is 0 and string is empty.
ResourceName	Resource name.
EventTypeId	Event type ID.
EventTypeName	Event type name.
ServiceId	Service ID.
ServiceName	Service name.
ContractPartyId	Contract Party ID.
ContractPartyName	Contract party name.
TimeZoneId	Time zone ID.
TimeZoneName	Time zone name.

Attribute	Description
TranslatedTo	Translated to the desired value.
CreatedByAdapterId	The ID of the adapter that created the translation entry.
CreatedByAdapterName	The name of the adapter that created the translation entry.
CreatedByUserId	The ID of the user that created the translation entry.
CreatedByUserName	The name of the user that created the translation entry.
LastActionDate	Date of the last action. Date that the last action (e.g., add / translate /ignore /delete) was done to the translation entry.
CreateDate	Date the translation entry was created.
ModifyDate	Date the translation entry was last modified. Same as LastActionDate.
Dump()	Returns string with the translation entry details.

AllFieldsValues(Delimiter)	Delimiter character that will be inserted between the field values.
	This method returns string with all values separated by the given delimiter. Example:
	`Sub OnTranslationEvent(EntryDetails)`
	`If EntryDetails.TableName <> "resourceTable" Then`
	`Tools.Log EntryDetails.Dump`
	`Else`
	`Tools.log EntryDetails.TableName & " (" & EntryDetails.TableId & ") > " & EntryDetails.AllFieldsValues("-")`
	`End If`
	`End Sub`

Safe ODBC

This safe component lets the user connect to any database using ODBC for reading only.

Close

Description

Closes the connection.

Syntax

Close

Open

Description

Opens the database for reading using a connection string.

Syntax

Open (ConnectionString)

Open Insight

Description

Opens the Insight database for reading without using a connection string.

Syntax

OpenInsight

SafeODBC Example

```
 Dim safeodbc
  Dim name
  Dim recordSet
  Set safeodbc=CreateObject("SafeOdbc.OdbcConnection")
  safeodbc.OpenOblicore
  Set recordSet= safeodbc.Query("select * from T_resources")
   Do Until recordSet.EOF
      Tools.log "Resource name = " & recordSet ("RESOURCE_NAME")
      recordSet.MoveNext
   Loop
recordSet.Close
  safeodbc.Close
```

Query

Description

This method gets the select statement and returns a record set.

Syntax

Query (SelectStatement)

Tools

The Tools component supplies all of the methods required for handling Translation entries and adding Resources, Event types, Services, and Contract parties.

Some important information about the Tools component parameters:

- Methods in the Tools component get and return object names, except for the translation entry methods.
- Object lists are converted to maps. Each object is one entry in the map that contains a map of fields.
- Each field is one entry, where the key is the field's name and the value is the field's value.
- List or group of objects, such as resource types attached to a resource, are represented as maps. When the value contains a string, the key is equal to the value.
- The date and time values, in the parameters and in the result, use the time zone of the script and the date format of the time zone.
- All parameters are mandatory, unless explicitly noted as optional.

Attach & Detach Methods

The following methods receive as a parameter the name of a change set. This parameter sets the date on which changes are performed to the given resources, and attaches these changes to the change set. The date of the change is taken from the default date of the change set.

Only real changes are considered as changes. Attaching a resource to a service on a certain date is considered as a change only if it was not attached on the same date previously. No error message is given for 'non-real' attachments or disconnections.

No changes can be performed on a resource or resource group that contains uncommitted changes that are attached to another change set and/or are on different dates.

AttachResourcesToContractParties

Description

This method attaches all the resources and resource groups in the first map to the contract parties in the second map.

Syntax

AttachResourcesToContractParties (Resources, ChangeSet, ContractParties)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
ContractParties	string/map	Name of contract party, or map of contract parties. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given contract parties must be defined.

AttachResourcesToResourceGroups

Description

This method attaches all the resources and resource groups in the first map to the resource groups in the second map.

Syntax

AttachResourcesToResourceGroups (Resources, ChangeSet, ResourceGroups)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	

Name	Type	Description
ResourceGroups	string/map	Name of resource group or map of resource groups. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given resource groups must be defined.

AttachResourcesToResourceTypes

Description

This method attaches all the resources and resource groups in the first map to the resource types in the second map.

Syntax

AttachResourcesToResourceTypes (Resources, ChangeSet, ResourceTypes)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
ResourceTypes	string/map	Name of resource type or map of resource types. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given resource types must be defined.

AttachResourcesToServices

Description

This method attaches all the resources and resource groups in the first map to the services in the second map.

Syntax

AttachResourcesToServices (Resources, ChangeSet, Services)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
Services	string/map	Name of service, or map of service. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given services must be defined.

DetachResourcesFromAllContractParties

Description

This method detaches all the resources and resource groups in the first map from the contract parties in the second map.

Syntax

DetachResourcesFromAllContractParties (Resources, ChangeSet)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	

Remarks

All the given resources and resource groups must be defined.

All the given services must be defined.

DetachResourcesFromAllResourceGroups

Description

This method detaches all the resources and resource groups in the first map from the resource groups in the second map.

Syntax

DetachResourcesFromAllResourceGroups (Resources, ChangeSet)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	

Remarks

All the given resources and resource groups must be defined.

All the given resource groups must be defined.

DetachResourcesFromAllServices

Description

This method detaches all the resources and resource groups in the first map from the services in the second map.

Syntax

DetachResourcesFromAllServices (Resources, ChangeSet)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.

Name	Type	Description
ChangeSet	string	

Remarks

All the given resources and resource groups must be defined.

All the given services must be defined.

DetachResourcesFromContractParties

Description

This method detaches all the resources and resource groups in the first map from the contract parties in the second map.

Syntax

DetachResourcesFromAllContractParrties (Resources, ChangeSet, ContractParties)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
ContractParties	string/map	Name of contract party, or map of contract parties. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given contract parties must be defined.

DetachResourcesFromResourceGroups

Description

This method detaches all the resources and resource groups in the first map from the resource groups in the second map.

Syntax

DetachResourcesFromResourceGroups (Resources, ChangeSet, ResourceGroups)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
ResourceGroups	string/map	Name of resource group or map of resource groups. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given resource groups must be defined.

DetachResourcesFromResourceTypes

Description

This method detaches all the resources and resource groups in the first map from the resource types in the second map.

Syntax

DetachResourcesFromResourceTypes (Resources, ChangeSet, ResourceTypes)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	

Name	Type	Description
ResourceTypes	string/map	Name of resource type, or map of resources types. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given resource types must be defined.

The resource should stay with at least one resource type.

DetachResourcesFromServices

Description

This method detaches all the resources and resource groups in the first map from the services in the second map.

Syntax

DetachResourcesFromServices (Resources, ChangeSet, Services)

Parameters

Name	Type	Description
Resources	string/map	Name of resource or resource group, or map of resources and resource groups. Map cannot be empty.
ChangeSet	string	
Services	string/map	Name or map of service. Map cannot be empty.

Remarks

All the given resources and resource groups must be defined.

All the given services must be defined.

Change Set Methods

AddChangeSet

Description

This method adds a new change set and returns the new change set ID.

Syntax

AddChangeSet (ChangeSetName, defaultDate, Description)

Parameter

Name	Type	Description
ChangeSetName	string	The name of the change set.
defaultDate	string	Represents the date in the script's time zone. It can also contain hours, minutes, and seconds.
Description	string	

Remarks

The change is added in Enable status.

GetChangeSetDetails

Description

This method returns a map with the change set details.

Syntax

GetChangeSetDetails (ChangeSetName)

Parameters

Name	Type	Description
ChangeSetName	string	The name of the change set.

Return Value

A map containing the following entries:

- ChangeSetName
- ChangeSetDefaultDate
- ChangeSetDescription
- ChangeSetStatus

Remarks

This method fails if the change set does not exist.

IsChangeSetExists

Description

This method checks whether the specified change set exists.

Syntax

IsChangeSetExists (ChangeSetName)

Parameters

Name	Type	Description
ChangeSetName	string	The name of the change set.

Return Value

True if the change set exists, **False** if it does not.

Remarks

None.

SearchChangeSets

Description

This method returns a map of change sets, ordered by the effective time.

Syntax

SearchChangeSets (defaultDate, ChangeSetNamePattern)

Parameters

Name	Type	Description
defaultDate	string	Represents the date in the script's time zone. It can also contain hours, minutes, and seconds.
ChangeSetNamePattern	string	Can use wildcards.

Remarks

All parameters are strings. Parameters are optional and the default values are empty strings. An empty string means that the parameter does not participate in the search.

The **defaultDate** parameter is checked against the change set's default date.

UpdateChangeSet

Description

This method updates the change set with the new details.

Syntax

UpdateChangeSet (OldChangeSetName, NewChangeSetName, defaultDate, Description)

Parameters

Name	Type	Description
OldChangeSetName	string	The name of the change set that needs to be changed.
NewChangeSetName	string	The new name of the change set.
defaultDate	string	Represents the date in the script's time zone. It can also contain hours, minutes, and seconds.

Name	Type	Description
Description	string	

Remarks

The values for **OldChangeSetName** and **NewChangeSetName** must be the same if the name does not change.

Contract Party Methods

AddContractPartyByMap

Description

This method adds the new contract party and returns the new contract party ID.

Syntax

AddContractPartyByMap (ContractPartyDetails)

Parameters

Name	Type	Description
ContractPartyDetails	map	Map of contract party details, as returned from GetContractPartyDetails (see page 53). The ContractPartyGroups and the UserGroups entries can be null, string, or map.

Remarks

None.

GetContractPartyDetails

Description

This method returns a map with the contract party details.

Syntax

GetContractPartyDetails (ContractPartyName)

Parameters

Name	Type	Description
ContractPartyName	string	The name of the contract party.

Return Value

A map containing the following entries:

- ContractPartyName
- ContractPartyDescription
- ContractPartyType
- ContractPartyRegistrationDate
- ContractPartyAddress
- ContractPartyCountry
- ContractPartyState
- ContractPartyStateZipcode
- ContractPartyContact
- ContractPartyPhoneNumber1
- ContractPartyPhoneNumber2
- ContractPartyFaxNumber
- ContractPartyMailAddress
- ContractPartyNotes
- ContractPartyClass
- ContractPartySeats
- ContractPartyGroups – Map of contract party groups.
- UserGroups – Map of users and user groups.

Remarks

Each entry in the ContractPartyGroups map contains the contract party group name in the key and in the value.

Each entry in the UserGroups map contains the user group name in the key and in the value.

IsContractPartyExists

Description

This method checks whether the specified contract party exists.

Syntax

IsContractPartyExists (ContractPartyName)

Parameters

Name	Type	Description
ContractPartyName	string	The name of the contract party.

Return Value

True if the contract party exists, **False** if it does not.

Remarks

None.

SearchContractParties

Description

This method returns a map of the contract parties.

Syntax

SearchContractParties (ContractPartyNamePattern, ContractPartyDescriptionPattern)

Parameters

Name	Type	Description
ContractPartyNamePattern	string	

Name	Type	Description
ContractPartyDescriptionPattern	string	

Remarks

All parameters are optional. Each entry in the map contains the contract party name in the key and in the value.

UpdateContractPartyByMap

Description

This method updates the contract party by map.

Syntax

UpdateContractPartyByMap (ContractPartyName, ContractPartyDetails)

Parameters

Name	Type	Description
ContractPartyName	string	The name of the contract party.
ContractPartyDetails	map	Map of the contract party group. The details are as in AddContractPartyByMap (see page 52).

Remarks

None.

Contract Party Group Methods

AddContractPartyGroupByMap

Description

This method adds a contract party group and returns the new contract party group ID.

Syntax

AddContractPartyGroupByMap (ContractPartyGroupDetails)

Parameters

Name	Type	Description
ContractPartyGroupDetails	map	Map of contract party. The details are similar to the map that is returned from GetContractPartyGroupDetails (see page 56). ■ ContractPartyGroupName ■ ContractPartyGroupDescription ■ Users: Null, one user, or map of users and user groups. The map can be empty. ■ ContractParties: Null, one contract party, or map of contract parties. The map can be empty.

Remarks

None.

GetContractPartyGroupDetails

Description

This method returns a map with the contract party group details.

Syntax

GetContractPartyGroupDetails (ContractPartyGroupName)

Parameters

Name	Type	Description
ContractPartyGroupName	string	The contract party group name. The maximum string length is 60 characters.

Return Value

A map containing the following entries:

- ContractPartyGroupName
- ContractPartyGroupDescription
- Users: Map of users and user groups. The map can be empty.
- ContractParties: Map of attached contract parties. The map can be empty.

Remarks

None.

IsContractPartyGroupExists

Description

This method checks whether the specified contract party group exists.

Syntax

IsContractPartyGroupExists(ContractPartyGroupName)

Parameters

Name	Type	Description
ContractPartyGrroupName	string	The contract party group name. The maximum string length is 60 characters.

Return Value

True if the contract party group exists, **False** if it does not.

Remarks

Running the method on this group returns the value **True**. There is a default contract party group called **All**.

SearchContractPartyGroups

Description

This method returns a map of contract party groups.

Syntax

SearchContractPartyGroups(ContractPartyGroupNamePattern, ContractPartyGroupDescriptionPattern)

Parameters

Name	Type	Description
ContractPartyGroupNamePattern	string	
ContractPartyGroupDescriptionPattern	string	

Remarks

All parameters are optional. The default is an empty string, which means to ignore the parameter. Each entry in the map contains the contract party name in the key and in the value. The **All** group can be in the map.

UpdateContractPartyGroupByMap

Description

This method updates the contract party group by map.

Syntax

UpdateContractPartyGroupByMap (ContractPartyGroupName, ContractPartyGroupDetails)

Parameters

Name	Type	Description
ContractPartyGroupName	string	The contract party group name. The maximum string length is 60 characters.
ContractPartyGroupDetails	map	Map of the contract party. The details are as in AddContractPartyGroupByMap (see page 56).

Remarks

None.

Domain Category Methods

GetDomainCategoryDetails

Description

This method returns a map with the domain category details.

Syntax

GetDomainCategoryDetails (DomainCategoryName)

Parameters

Name	Type	Description
DomainCategoryName	string	The domain category name. The maximum string length is 60 characters.

Return Value

A map containing the following entries:

- DomainCategoryName
- DomainCategoryDescription

Remarks

None.

IsDomainCategoryExists

Description

This method checks whether the specified domain category exists.

Syntax

IsDomainCategoryExists(DomainCategoryName)

Parameters

Name	Type	Description
DomainCategoryName	string	The domain category name. The maximum string length is 60 characters.

Return Value

True if the domain category exists, **False** if it does not.

Remarks

None.

SearchDomainCategories

Description

This method returns a map of domain categories.

Syntax

SearchDomainCategories (DomainCategoriesNamePattern)

Parameters

Name	Type	Description
DomainCategoriesNamePattern	string	

Remarks

All parameters are optional. The default is an empty string, which means to ignore the parameter. Each entry in the map contains the domain category name in the key and in the value.

Commit Change Set Methods

CommitChangeSets

Description

This method commits all the resources with uncommitted changes in the given change set(s).

Syntax

CommitChangeSet (ChangeSets)

Parameters

Name	Type	Description
ChangeSets	string/map	The name of a change set or map of change sets. The map cannot be empty.

Remarks

This method uses time-consuming operations, whose duration depends not only on the number of uncommitted changes that the operations are performed upon, but also on the size of the resource tables. Therefore, CA recommends performing this operation on a number of changes, and not on each change separately.

Event Type Methods

AddEventTypeByMap

Description

This method adds a new event type and returns the new event type ID.

Syntax

AddEventTypeByMap (EventTypeDetails)

Parameters

Name	Type	Description
EventTypeDetails	map	Map of event type details, as returned from GetEventTypeDetails (see page 62). The Adapters and ResourceTypes entries can be null, string, or map.

Remarks

None.

GetEventTypeDetails

Description

This method returns a map with the event type details.

Syntax

GetEventTypeDetails (EventTypeName)

Parameters

Name	Type	Description
EventTypeName	string	The event type name.

Return Value

A map containing the following entries:

- EventTypeName
- EventTypeDescription
- EventTypesFields – Each entry in this map contains field details. The key is the field sequence number starting with zero. The entries include:
  - EventTypeFieldSequenceNumber
  - EventTypeFieldName
  - EventTypeFieldType
- ResourceTypes (map). Map of all related resource types.
- Adapters (map). Map of related adapters.

Remarks

None.

IsEventTypeExists

Description

This method checks whether the specified event type exists.

Syntax

IsEventTypeExists (EventTypeName)

Parameters

Name	Type	Description
EventTypeName	string	The event type name.

Return Value

True if the event type exists, **False** if it does not.

Remarks

None.

Exception Methods

AddException

Description

This method adds a new exception and returns a map with the following entries:

- ExceptionName
- ExceptionJustification
- ExceptionDescription
- ExceptionEffectiveFrom
- ExceptionEffectiveTo
- ExceptionTimeSlot
- ContractParties: Map of contract parties and/or contracts. The map can be empty.
- Contracts: (key ContractPartyNameContractName, Value ContractName)
- Services: Map of services. The map can be empty.
- DomainCategories: Map of domain categories. The map can be empty.

Note: The *Tools.AddException* parameter "ExceptionTimeSlot" should refer to the name of an existing "Timeslot Template"
(see Framework > Template Library > Timeslot Templates).

Note: This is different than the the GUI, which requires a timeslot **type** (Weekly/Yearly) (with an "Edit Timeslot" button), a timeslot **range** and a **time zone**.

Syntax

AddException (pDetails)

Parameters

Name	Type	Description
pDetails	map	Map of exception details.

Remarks

None.

Example:

```
Option Explicit

Sub OnLoad()
  'Tools.log "Translation Script : In OnLoad procedure", "I"
End Sub

Sub OnTranslationEvent(EntryDetails)
  'Tools.log "Translation Script : In OnTranslationEvent function", "I"
End Sub

Sub Main()
  'Tools.log "Translation Script : In Main procedure", "I"

  Dim map2
  Set map2=Tools.CreateMap

  map2("Exception1")="Exception2"
  map2("ExceptionName")="Exception2"
  map2("ExceptionJustification")="Exception2"
  map2("ExceptionDescription")=""
  map2("ExceptionEffectiveFrom")="01/12/2009"
  map2("ExceptionEffectiveTo")="05/12/2009"
  map2("ExceptionTimeSlot")="Always"
  map2("ContractParties") = Tools.CreateMap
  'Uncomment this line to assign the exception to contract party: contractParty1
  'map2("ContractParties")("contractPartyName")="contractParty1"
  map2("Contracts")= Tools.CreateMap
  'Comment out this line if only assigning the exception to a contract party, and not
a contract
  'The following key is in the format: ContractPartyNameContractName
  map2("Contracts")("contractParty1contract1")= "contract1"
  map2("Services")=Empty
  map2("DomainCategories")=Empty

Tools.AddException(map2)

Tools.Commit

End Sub

Function Result
  'Tools.log "Translation Script : In Result function", "I"
  'Result =
```

```
End Function
```

DeactivateException

Description

This method enables the user to deactivate an active exception.

Syntax

DeactivateException (wsExceptionName, wsJustification)

Parameters

Name	Type	Description
wsExceptionName	string	
wsJustification	string	

Remarks

None.

SearchExceptions

Description

This method returns a map of exceptions with the ExceptionName.

Syntax

SearchExceptions(pDetails)

Parameters

Name	Type	Description
pDetails	map	

Remarks

None.

General Methods

Commit

Description

This method commits the current transaction and opens a new transaction.

Syntax

Commit ()

Remarks

When the translation script is run in the script scope, the Commit method is not executed in the database (the database transaction is not committed).

A translation script automatically opens a new transaction when it starts and immediately after the Commit action.

If the user does not commit the transaction (due to an error or some other reason) the script is automatically rolled back.

CreateMap

Description

This method returns a new map.

Syntax

CreateMap ()

Return Value

A new empty map.

Remarks

Use this method when you need a new empty map (instead of CreateObject("SlalomMap.map")).

IsTimeZoneExists

Description

This method checks whether the specified time zone exists.

Syntax

IsTimeZoneExists (TimeZoneName)

Parameters

Name	Type	Description
TimeZoneName	string	The name of the time zone.

Return Value

True if the time zone exists, **False** if it does not.

Remarks

None.

Log

Description

This method writes the input string to the log table.

Syntax

Log (String, Level)

Parameters

Name	Type	Description
Log	String	The Level parameter is optional. Legal values are I, W, E, and D. The default is I.

Remark

Legal values are:

- **I**: Info
- **W**: Warning
- **E**: Error
- **D**: Debug

Messages with D level are written in debug mode only and appear as information messages.

Rollback

Description

This method rolls back the current transaction and opens a new transaction.

Syntax

Rollback ()

Remarks

When the translation script is run in the script scope, the Rollback method is not executed in the database (the database transaction is not rolled backed).

A translation script automatically opens a new transaction when it starts and immediately after the Rollback action.

If the user does not commit the transaction (due to an error or for another reason) the script is automatically rolled back.

Organization Methods

AddOrganization

Description

This method adds an organization.

Syntax

AddOrganization (OrganizationName, OrganizationDescription)

Parameters

Name	Type	Description
OrganizationName	string	The name of the organization.
OrganizationDescription	string	A description of the organization.

Remarks

None.

GetOrganizationName

Description

This method returns the organization name from the user full name.

Syntax

GetOrganizationName (UserFullName)

Parameters

Name	Type	Description
UserFullName	string	The user full name.

Return Value

A string, starting with first character after the first '@' up to the end of the string. If the '@' character does not exist, it returns an empty string.

Remarks

This method does **not** check if the user exists or if the full name is legal.

IsOrganizationExists

Description

This method checks whether the specified organization exists.

Syntax

IsOrganizationExists(OrganizationName)

Parameters

Name	Type	Description
OrganizationName	string	The name of the organization.

Return Value

True if the organization exists, **False** if it does not.

Remarks

None.

Resource Methods

The following methods receive as a parameter the name of a change set. This parameter sets the date on which changes are performed to the given resources, and attaches these changes to the change set. The date of the change is taken from the default date of the change set.

Only real changes are considered as changes. Attaching a resource to a service on a certain date is considered as a change only if it was not attached on the same date previously. No error message is given for 'non-real' attachments or disconnections.

No changes can be performed on a resource or resource group that contains uncommitted changes that are attached to another change set and/or are on different dates.

AddResource

Description

This method adds a new resource and returns the new resource group ID.

This method sets the first effective time (in the resource) to the change set's default date.

Syntax

AddResource (ResourceName, ChangeSetName, ResourceDescription, ResourceTypes, ResourceGroups, Services, ContractParties)

Parameters

Name	Type	Description
ResourceName	string	Name of the resource.
ChangeSetName	string	Name of the change set.
ResourceDescription	string	Description of the resource. Can be empty.
ResourceTypes	string/map	Name of a resource type or map of resource types. The map cannot be empty.
ResourceGroups	(string/map/null)	Name of a resource group, map of resource groups (can be empty), or null.
Services	string/map/null	Name of a service, map of services (can be empty), or null.
ContractParties	(string/map/null)	Name of a contract party, map of contract parties (can be empty), or null.

Remarks

All input objects must exist. This method fails if it is missing an input object.

AddResourceByMap

Description

This method adds a new resource and returns the new resource ID.

This method sets the first effective time (in the resource) to the change set's default date.

Syntax

AddResourceByMap (ResourceDetails)

Parameters

Name	Type	Description
ResourceDetails	map	Map of resource details similar to the map returned from the GetResourceDetails method. The ResourceTypes entry must contain at least one resource type. The ResourceGroups, Resources and Services internal maps can be empty. The CustomAttributes internal map must contain all the custom attributes attached to resource. The ResourceGroupId entry is not relevant.

Remarks

None.

Example

```
Sub Main()
  Dim strResource
  strResource = "resource1"

  If Tools.IsResourceExists(strResource) Then
    Tools.log "Resource already exists - change resource name", "I"
  Else
    Dim ResourceMap
    Set ResourceMap= Tools.CreateMap

    ResourceMap("ResourceName") = strResource
    ResourceMap("ResourceDisplayName") = strResource
    ResourceMap("ChangeSetName") = "Default"
    ResourceMap("ResourceDescription") = "Added automaticaly by script"
    ResourceMap("ResourceTypes") = "Default"
    ResourceMap("ResourceGroups") = Null
    ResourceMap("Services") = Null
    ResourceMap("ContractParties") = Null
    ResourceMap("Effective") = "Yes"
    ResourceMap("CustomAttributes") = Null
    ResourceMap("ResourceTimeZone") = "GMT"

    Tools.AddResourceByMap ResourceMap
    Tools.Commit

    Dim mapGenDetails
    Set mapGenDetails = Tools.GetResourceGeneralDetails(strResource)
    Tools.Log mapGenDetails.dump

  End If
End Sub
```

CommitResources

Description

This method commits the given resources. The parameter contains the name of a resource, resource group, or map of resources and resource groups.

Syntax

CommitResources (Resources)

Parameters

Name	Type	Description
Resources	string/map	Name of a resource, resource group, or map of resources and/or resource groups. The map cannot be empty.

Remarks

This method updates the database, but does not perform the database commit. The Tools.Commit method must be used in the script as well.

This method uses time-consuming operations, whose duration depends not only on the number of uncommitted changes that the operations are performed upon, but also on the size of the resource tables. Therefore, CA recommends performing this operation on a number of changes, and not on each change separately.

GetResourceDetails

Description

This method returns a map with the resource details.

Syntax

GetResourceDetails (ResourceName, ChangeSetName, CommittedOnly)

Parameters

Name	Type	Description
ResourceName	string	The name of the resource.
ChangeSetName	string	The name of the change set.

Name	Type	Description
CommittedOnly	boolean	This parameter is optional. The default is **False**. When **True**, only committed values are retrieved. When **False**, get value and consider uncommitted values.

Return Value

A map containing the following entries:

- ChangeSetName
- ContractParties: Map of contract parties. The map can be empty.
- CustomAttributes: Map of custom attributes. Each entry in this map is a map that defines one custom attribute. The key of each entry is the custom attribute name, and the value is an internal map with the following two entries:
  - CustomAttributeName: The custom attribute name.
  - CustomAttributeValue: The custom attribute value.
- Effective: Yes or No. Defines if the resource group is effective in the specific date.
- ResourceDescription
- ResourceDisplayName
- ResourceGroups: Map of resource groups. The Map can be empty.
- ResourceId: The resource group ID.
- ResourceName
- ResourceTimeZone
- ResourceTypes: Map of resource types.
- Services: Map of services. The map can be empty.

Remarks

Each entry in each internal map contains the object name in the key and in the value.

This method fails if the resource and/or the change set do not exist.

GetResourceDetailsByDate

Description

The method returns a map with the resource details.

This method returns the resource details true to the given date. If the third parameter is false, it contains uncommitted changes as well. The returned map is the resource details map as given by *GetResourceDetails* (see page 75).

Syntax

GetResourceDetailsByDate (ResourceName, Date, CommittedOnly)

Parameters

Name	Type	Description
ResourceName	string	The name of the resource.
Date	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.
CommittedOnly	Boolean	This parameter is optional. The default is **False**. When **True**, only committed values are retrieved. When **False**, get value and consider uncommitted values.

Return Value

See GetResourceDetails (see page 75)

Note: GetResourceDetailsByDate returns the values of the function GetResourceGroupDetails, but without the ChangeSetName.

Remarks

Each entry in each internal map contains the object name in the key and in the value.

This method fails if the resource and/or the change set do not exist.

GetResourceGeneralDetails

Description

This method returns a map with the resource general details.

Syntax

GetResourceGeneralDetails (ResourceName)

Parameters

Name	Type	Description
ResourceName	string	The name of the resource.

Return Value

A map containing the following entries:

- ResourceDescription
- ResourceDisplayName
- ResourceId: The Resource Group ID.
- ResourceName

The general details are not version controlled.

Remarks

This method fails if the resource does not exist.

GetResourceChildrenList

Description

This method uses the Resources cache to gather information about any given resource at a given time stamp.

For example, given "Resource Group A", this method will tell us this method will return us a slalom map containing the resources which directly **descend** from it **and** are effective **and** the connection between them is also effective at the given TIME.

Syntax

GetResourceChildrenList(resourceGroupId,TIME)

Parameters

Name	Type	Description
ResourceGroupID	string	The name of the resource.
TIME	string	Represents a date in the script's time zone.

Return Value

Returns a slalom map with all the direct descendants of the given ResourceGroupId at the given time.

Remarks

This method fails if the resource does not exist.

Slalom Example (partial)

```
Option Explicit

Sub OnRegistration(dispatcher)
  'TODO: ADD code here TO REGISTER EVENTS NEEDED TO calculate service LEVEL
End Sub

Sub OnResourceStructureChanged(TIME)
  Set pMap =
Context.GetResourceChildrenList(Context.ResourceGroupId("xxxxR"),TIME)
  for each pItem in pMap
   \\some code
  Next
End Sub

Sub OnMetricChanged(TIME)
'TODO: this code is executed at the beginning of every contract version
End Sub

Sub OnLoad(TIME)
  'TODO: ADD code here TO handle calculation START event
End Sub

Sub OnPeriodStart(TIME)
  'TODO: ADD code here TO handle period START event
End Sub

Sub OnPeriodEnd(TIME, isComplete)
  'TODO: ADD code here TO handle period END event
End Sub
```

GetResourceParentsList

Description

This method uses the Resources cache to gather information about any given resource at a given time stamp.

For example, given "Resource Group A", this method will return us a slalom map containing the resources which directly **ascend** from it **and** are effective **and** the connection between them is also effective at the given TIME.

Syntax

GetResourceParentsList(ResourceId,TIME)

Parameters

Name	Type	Description
ResourceID	string	The name of the resource.
TIME	string	Represents a date in the script's time zone.

Return Value

Returns a slalom map with all the direct **ascendants** of the given ResourceId at the given time.

Remarks

This method fails if the resource does not exist.

Slalom Example (partial)

```
Option Explicit

Sub OnRegistration(dispatcher)
  'TODO: ADD code here TO REGISTER EVENTS NEEDED TO calculate service LEVEL
End Sub

Sub OnResourceStructureChanged(TIME)
  Set pMap = Context.GetResourceParentsList(Context.ResourceId("xxxxR"),TIME)
  for each pItem in pMap
   \\some code
  Next
End Sub

Sub OnMetricChanged(TIME)
'TODO: this code is executed at the beginning of every contract version
End Sub

Sub OnLoad(TIME)
  'TODO: ADD code here TO handle calculation START event
End Sub

Sub OnPeriodStart(TIME)
  'TODO: ADD code here TO handle period START event
End Sub

Sub OnPeriodEnd(TIME, isComplete)
  'TODO: ADD code here TO handle period END event
End Sub
```

IsResourceExists

Description

This method checks whether the specified resource exists.

Syntax

IsResourceExists (ResourceName)

Parameter(s)

Name	Type	Description
ResourceName	string	The name of the resource.

Return Value

True if the resource exists, **False** if it does not.

Remarks

None.

IsResourceExistsInTime

Description

This method checks whether the resource committed as effective is in the given timeframe.

Syntax

IsResourceExistsInTime (ResourceName, EffectiveDate)

Parameters

Name	Type	Description
EffectiveDate	string	Represents the date in the script's time zone. It can contain hours, minutes, and seconds.
ResourceName	string	The name of the resource.

Return Value

True if the resource committed as effective is in the given timeframe, **False** if it is not.

Remarks

This method fails if the resource does not exist.

IsResourceExistsInResourceVersion

Description

This method checks whether the resource committed as effective is in the timeframe from the given change set.

Syntax

IsResourceExistsInResourceVersion (ResourceName, ChangeSetName)

Parameters

Name	Type	Description
ResourceName	string	The name of the resource.
ChangeSetName	string	The name of the change set.

Return Value

True if the resource committed as effective is in the timeframe from the given change set, **False** if it is not.

Remarks

This method fails if the resource does not exist.

SearchResources

Description

This method searches resources that are committed as effective in the given date.

Syntax

SearchResources (EffectiveDate, ResourceNamePattern)

Parameters

Name	Type	Description
EffectiveDate	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.
ResourceNamePattern	string	Can use wildcards.

Return Value

A map of resources, where the key and the value of each entry is the resource name.

Remarks

All the parameters are optional.

The default is an empty string, which means to ignore the parameter.

If the effective date is empty, every resource group that is committed is returned as effective at some point in time.

UpdateResourcesCustomAttribute

Description

This method updates the value of the custom attribute in the specific resources in a given date.

Syntax

UpdateResourcesCustomAttribute (Resources, ChangeSetName, CustomAttributeName, CustomAttributeValue)

Parameters

Name	Type	Description
Resources	string/map	Name of a resource or resource group, or map of resources and/or resource groups. The map cannot be empty.
ChangeSetName	string	The name of the change set.
CustomAttributeName	string	The custom attribute name as the entry key.
CustomAttributeValue	string	The new custom attribute's value as the entry value.

Remarks

This method fails if any of the following is true:

- A resource/resource group does not exist.
- The change set does not exist.
- A custom attribute does not exist or is not attached to all the resources/resource groups.
- A custom attribute values is not legal.

UpdateResourcesEffective

Description

This method updates the effective field in the given resources in a given date.

The optional *EffectiveDate* parameter maintains backward compatibility with existing scripts.

Note: Setting the effective date overrides the default behavior, which is to take the effective date from the change set.

Syntax

UpdateResourcesEffective (Resources, ChangeSetName, IsEffective,EffectiveDate)

Parameters

Name	Type	Description
Resources	string/map	Name of a resource or resource group, or map of resources and/or resource groups. The map cannot be empty.
ChangeSetName	string	The name of the change set.
IsEffective	Boolean	**True** or **False**.
EffectiveDate [optional, defaultvalue=""]	string	The date from which CA CA Business Service Insight may use the resource. The resource effective date (in the script's time zone) is defined by the resource version that includes the resource. The date can contain hours, minutes and seconds.

Remarks

This method fails if either of the following is true:

- A resource/resource group does not exist.
- The change set does not exist.

UpdateResourceByMap

Description

This method updates a resource detail on the effective date defined within the change set.

Syntax

UpdateResourceByMap (ResourceName, ResourceDetails)

Parameters

Name	Type	Description
ResourceName	string	The name of the resource.
ResourceDetails	Map	Map of resource details is in the structure of the map returned from GetResourceGroupDetails method (including returns ResourceTimeZone). The ResourceGroups, Resources, and Services internal maps can be empty. The ResourceGroupId entry is not relevant. The ResourceTypes entry must contain at least one resource type. The CustomAttributes internal map must contain all of the custom attributes that are attached to the resource.

Remarks

We recommend calling "Tools.UpdateResourceByMap" (or "Tools.UpdateResourceGroupByMap") instead of multiple calls to "Tools.UpdateResourcesCustomAttribute".

Example

```
Option Explicit

Const CHANGESET_NAME = "Default"
'This script assumes that the following resource exists
Const RESOURCE_TEMPLATE_NAME = "ResourceTemplate"

Sub OnLoad()
'Tools.log "Integration Script: In OnLoad procedure", "I"

End Sub

Sub Main()
  Dim resourceName
  resourceName = "MyResource1"
  Dim aMap
  Set aMap =  Tools.GetResourceDetails (RESOURCE_TEMPLATE_NAME, CHANGESET_NAME)

  aMap("ResourceName") = resourceName
  'aMap("ChangeSetName") = CHANGESET_NAME
  aMap("CustomAttributes")("Owner")("CustomAttributeValue") = "Joe"
  aMap("CustomAttributes")("SerialNumber")("CustomAttributeValue") = "123456"

  If Not Tools.IsResourceExists(resourceName) Then
            Tools.AddResourceByMap(aMap)
      Else
            Tools.UpdateResourceByMap resourceName, aMap
  End If

  'Tools.CommitResources resourceName
  Tools.Commit

End Sub

Function Result
  'Tools.log " Integration Script : In Result function", "I"
  'Result =

End Function
```

UpdateResourceGeneralDetails

Description

This method updates the resource general details.

This update does *not* need to be committed by the *CommitResources* or *CommitChangeSets* methods.

Syntax

UpdateResourceGeneralDetails (OldResourceName, ResourceGeneralDetails)

Parameters

Name	Type	Description
OldResourceName	string	The name of the resource before the update.
ResourceGeneralDetails	Map	Map with the new general details, as returned from *GetResourceGeneralDetails* (see page 78). The user can change any field in the map.

Remarks

This method fails if the resource group does not exist.

Bulk Methods

The usage of the methods is different than the existing translation script methods, and therefore they are prefixed with the term Bulk.

Each method obtains input parameters in two manners:

- Via standard input parameters. One of these parameters is an integer called session-Id (mentioned below).
- The user populates certain user database tables (prefixed with T_API) within the CA Business Service Insight database before calling one of the Bulk methods.

The schema of the T_API tables is maintained by CA Business Service Insight, but their content is maintained by the end user.

Each table is used in exactly one translation script method (although one translation script method can use many user database tables).

Each table has a session-Id column for supporting running two scripts simultaneously with different session-Ids. The session-Id column defaults to zero.

The user can populate the T_API tables via the translation script before calling the new translation script methods or within a separate external process before calling the translation script..

The T_API tables mentioned here do not reference other CA Business Service Insight tables.

The T_API tables may have primary key or uniqueness constraints, and they may have foreign key constraints to other T_API tables for the same method.

BulkUpdateResourceGroup

Referenced Tables

T_API_UPDATE_RG

Resource Groups to create/update:

- RESOURCE_GROUP_NAME VARCHAR2(100) not null
- RESOURCE_GROUP_DESCRIPTION VARCHAR2(512)
- RESOURCE_GROUP_DISPLAY_NAME VARCHAR2(100)
- SESSION_ID NUMBER default 0 not null

Primary key: (SESSION_ID, RESOURCE_GROUP_NAME)

T_API_UPDATE_RG_ATTRIBUTES

The custom attributes of each Resource Group:

- RESOURCE_GROUP_NAME VARCHAR2(100) not null
- CA_NAME VARCHAR2(60) not null
- CA_VALUE VARCHAR2(200)
- SESSION_ID NUMBER default 0 not null

Primary key: (SESSION_ID, RESOURCE_GROUP_NAME, CA_NAME)

Note: RESOURCE_NAME must correspond to an existing Resource Group names

Note: CA_NAME must correspond to existing Resource Group Custom Attribute names

Note: CA_VALUE must conform to type of the Custom Attribute (e.g. Text/Number/List)

Explicit Parameters

Parameter	Type	Description
sessionId	Integer	Only reference rows in the associated tables with the specified session-Id.
changeSetName	String	String Corresponds to an existing change set (e.g. “LV 2008”).

Note: This method ensures that all the resource groups mentioned exist within CA Business Service Insight with the updated custom attribute values.

Note: CA Business Service Insight resource groups not mentioned in the select statements are not affected.

Note: Cleaning up the tables T_API_UPDATE_RG and T_API_UPDATE_RG_ATTRIBUTES before (and optionally after) the method call is the responsibility of the translation script. This method can be called multiple times with the same data, only changing the referenced resources to “uncommitted resources” if there is an actual change to the specific resources custom attributes.

Limitations

- There is no checking for the existence of mandatory custom attributes. Therefore it is possible to create a resource with a mandatory custom attribute being not specified.
- Resource Groups Custom Attributes are expected to be of type “All Entities” and not “Selected types”

Example

T_API_UPDATE_RG

```
insert into T_API_UPDATE_RG
(
RESOURCE_GROUP_NAME,
RESOURCE_GROUP_DISPLAY_NAME
)
select
SAP_RESOURCENAME "RESOURCE_NAME",
SAP_DISPLAYNAME "RESOURCE_DISPLAY_NAME"
from sbb_contract_topologie_new
UNION
select
ISAP_RESOURCENAME,
ISAP_DISPLAYNAME
from sbb_contract_topologie_new
```

T_API_UPDATE_RG_ATTRIBUTES

```
insert into T_API_UPDATE_RG_ATTRIBUTES
(
RESOURCE_GROUP_NAME,
CA_NAME,
CA_VALUE
)
select SAP_RESOURCENAME,
to_char('Basis Service Name'),
SAP_BASIS_SERVICE_NAME
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('Station Code'),
SAP_STATIONCODE
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('Location'),
SAP_LOCATION
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('sap State'),
SAP_SAPSTATE
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('RFS Date'),
SAP_RFSDATE
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('SLA'),
SAP_SLA
from sbb_contract_topologie_new
UNION
select SAP_RESOURCENAME,
to_char('Accountable CI'),
SAP_ACCOUNTABLE_CI
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('Basis Service Name'),
ISAP_BASIS_SERVICE_NAME
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('Station Code'),
```

```
ISAP_STATIONCODE
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('Location'),
ISAP_LOCATION
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('sap State'),
ISAP_SAPSTATE
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('RFS Date'),
ISAP_RFSDATE
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('SLA'),
ISAP_SLA
from sbb_contract_topologie_new
UNION
select ISAP_RESOURCENAME,
to_char('Accountable CI'),
ISAP_ACCOUNTABLE_CI
from sbb_contract_topologie_new
```

Calling Method

```
Const SessionId="0"
Const ChangeSetName = "LV 2008"
Sub Cleanup_T_API_Tables()
Dim sql
sql = "delete T_API_UPDATE_RG where SESSION_ID = " & SessionId
ExecuteSQL(sql)
sql = "delete T_API_UPDATE_RG_ATTRIBUTES where SESSION_ID = " &
SessionId
ExecuteSQL(sql)
End Sub
Sub Main()
Cleanup_T_API_Tables()
Populate_T_API_UPDATE_RG()
Populate_T_API_UPDATE_RG_ATTRIBUTES()
Tools.BulkUpdateResourceGroup SessionId, ChangeSetName
Cleanup_T_API_Tables()
End Sub
```

BulkUpdateResourceTree

Referenced Tables

T_API_UPDATE_RESOURCE_ATTACH

The pairs to attach:

- RESOURCE_NAME VARCHAR2(100) not null
- RESOURCE_GROUP_NAME VARCHAR2(100) not null
- SESSION_ID NUMBER default 0 not null

Primary key: (SESSION_ID, RESOURCE_NAME, RESOURCE_GROUP_NAME)

T_API_UPDATE_RESOURCE_DETACH

The pairs to detach:

- RESOURCE_NAME VARCHAR2(100) not null
- RESOURCE_GROUP_NAME VARCHAR2(100) not null
- SESSION_ID NUMBER default 0 not null

Primary key: (SESSION_ID, RESOURCE_NAME, RESOURCE_GROUP_NAME)

Explicit Parameters

Parameter	Type	Description
sessionId	Integer	Only reference rows in the associated tables with the specified session-Id.
changeSetName	String	String Corresponds to an existing change set (e.g. “LV 2008”).

Note: Any rows mentioned in both T_API_UPDATE_RESOURCE_ATTACH and T_API_UPDATE_RESOURCE_DETACH are ignored.

Note: This method also detaches resources to resource groups according to the resource tree provided in T_API_UPDATE_RESOURCE_DETACH.

Note: This method attaches resources to resource groups according to the resource tree provided within T_API_UPDATE_RESOURCE_ATTACH.

Note: This method requires the resources and resource groups listed to already exist, and therefore is typically called after the BulkUpdateResouceGroup method (perhaps in the same translation script).

Note: Cleaning up the tables T_API_UPDATE_RESOURCE_DETACH and T_API_UPDATE_RESOURCE_ATTACH before/after the method call is the responsibility of the translation script.

Limitations

- Any resources & resource groups mentioned in T_API_UPDATE_RESOURCE_DETACH and T_API_UPDATE_RESOURCE_ATTACH (that are not ignored due to duplications) will be changed to “uncommitted resources”, even if the specified attachment (detachment) already exists (doesn’t exist).
- Cleaning up the referenced tables before (and optionally after) the method call is the responsibility of the translation script.

Example

T_API_UPDATE_RESOURCE_ATTACH

```
insert into T_API_UPDATE_RESOURCE_ATTACH
(
RESOURCE_NAME,
RESOURCE_GROUP_NAME
)
select
RESOURCE_NAME,
ISAP_RESOURCENAME
from sbb_contract_topologie_new
where Source = 'BATSUP P-KS-V'
UNION
select
ISAP_RESOURCENAME,
SAP_RESOURCENAME
from sbb_contract_topologie_new
where Source = 'BATSUP P-KS-V'
```

T_API_UPDATE_RESOURCE_DETACH

```
insert into T_API_UPDATE_RESOURCE_DETACH
(
RESOURCE_NAME,
RESOURCE_GROUP_NAME
)
select
RESOURCE_NAME,
ISAP_RESOURCENAME
from sbb_contract_topologie_new
where Source = '_old_BATSUP P-KS-V'
UNION
select
ISAP_RESOURCENAME,
SAP_RESOURCENAME
from sbb_contract_topologie_new
where Source = '_old_BATSUP P-KS-V'
```

Calling Method

```
Const SessionId="0"
Const ChangeSetName = "LV 2008"
Sub Cleanup_T_API_Tables()
Dim sql
sql = "delete T_API_UPDATE_RESOURCE_ATTACH where SESSION_ID = " &
SessionId
ExecuteSQL(sql)
sql = "delete T T_API_UPDATE_RESOURCE_ATTACH where SESSION_ID = " &
SessionId
ExecuteSQL(sql)
End Sub
Sub Main()
Cleanup_T_API_Tables()
Populate_T_API_UPDATE_RESOURCE_ATTACH()
Populate_T_API_UPDATE_RESOURCE_DETACH()
Tools.BulkUpdateResourceTree SessionId, ChangeSetName
Cleanup_T_API_Tables()
End Sub
```

BulkDeleteResourceGroup

Referenced Tables

T_API_DELETE_RG

The resource groups to delete:

- RESOURCE_GROUP_NAME VARCHAR2(100) not null
- SESSION_ID NUMBER default 0 not null

Primary key: (SESSION_ID, RESOURCE_GROUP_NAME)

Explicit Parameters

Parameter	Type	Description
sessionId	Integer	Only reference rows in the associated tables with the specified session-Id.
changeSetName	String	String Corresponds to an existing change set (e.g. “LV 2008”).

Note: In order to detach a resource group from member resource groups, the method BulkUpdateResourceTree should be used.

Note: This method is intended to be used after BulkUpdateResourceTree has been called to remove all the attached resources.

Note: This method is currently designed for deleting resource groups only, and only resource groups with no attachments (member or child).

Note: In a future version, this method may deprecated in favor of a more generic “BulkDeleteResource” method that cater for deleting resources and resource groups.

Limitations

The “forceDeletion” parameter is currently ignored (assumed to be false)

Example

T_API_DELETE_RG

```
insert into T_API_DELETE_RG
(
RESOURCE_GROUP_NAME
)
select
SAP_RESOURCENAME
from sbb_contract_topologie_new
where TASK = 'DELETED'
UNION
select
ISAP_RESOURCENAME
from sbb_contract_topologie_new
where TASK = 'DELETED'
```

Calling Method

```
Const SessionId="0"
Sub Cleanup_T_API_Tables()
Dim sql
sql = "delete T_API_DELETE_RG where SESSION_ID = " & SessionId
ExecuteSQL(sql)
End Sub
Sub Main()
Cleanup_T_API_Tables()
Populate_T_API_DELETE_RG()
Tools.BulkDeleteResourceGroup SessionId, False
Cleanup_T_API_Tables()
End Sub
```

Resource Group Methods

The following methods receive as a parameter the name of a change set. This parameter sets the date on which changes are performed to the given resources, and attaches these changes to the change set. The date of the change is taken from the default date of the change set.

Only real changes are considered as changes. Attaching a resource to a service on a certain date is considered as a change only if it was not attached on the same date previously. No error message is given for 'non-real' attachments or disconnections.

No changes can be performed on a resource or resource group that contains uncommitted changes that are attached to another change set and/or are on different dates.

AddResourceGroup

Description

This method adds a new resource group and returns the new resource group ID.

This method sets the first effective time (in the resource group) to the change set's default date.

Syntax

AddResourceGroup (ResourceGroupName, ChangeSetName, ResourceGroupDescription, Resources, Services, ContractParties)

Parameters

Name	Type	Description
ResourceGroupName	string	The name of the resource group.
ChangeSetName	string	The name of the change set.
ResourceGroupDescription	string	A description of the resource group. Can be empty.
Resources	string/map/null	Name of a resource or resource group, map of resources and resource groups (can be empty), or null.
Services	string/map/null	Name of a service, map of services (can be empty), or null.
ContractParties	string/map/null	Name of a contract party, map of contract party (can be empty), or null.

Return Value

The new resource group ID.

Remarks

All input objects must exist. This method fails if it is missing an input object.

AddResourceGroupByMap

Description

This method adds a new resource group and returns the new resource group ID.

This method sets the first effective time (in the resource group) to the change set's default date.

Syntax

AddResourceGroupByMap (ResourceGroupDetails)

Parameter(s)

Name	Type	Description
ResourceGroupDetails	map	Map of resource group details is in the structure of the map returned from GetResourceGroupDetails (see page 100). The ResourceGroups, Resources, and Services internal maps can be empty. The CustomAttributes internal map must contain all of the custom attributes that are attached to the resource groups. The ResourceGroupId entry is not relevant.

Remarks

None.

GetResourceGroupDetails

Description

This method returns a map with the resource group details.

Syntax

GetResourceGroupDetails (ResourceGroupName, ChangeSetName, CommittedOnly)

Parameters

Name	Type	Description
ResourceGroupName	string	The name of the resource group.

Name	Type	Description
ChangeSetName	string	The name of the change set.
CommittedOnly	boolean	This parameter is optional. The default is **False**. When **True**, only committed values are retrieved. When **False**, get value and consider uncommitted values.

Return Value

The map contains the following entries:

- ChangeSetName
- ContractParties: Map of contract parties. The map can be empty.
- CustomAttributes: Map of custom attributes. Each entry in this map is a map that defines one custom attribute. The key of each entry is the custom attribute name and the value is an internal map with the following two entries:
  - CustomAttributeName: The custom attribute name.
  - CustomAttributeValue: The custom attribute value.
- Effective: Yes or No. Defines if the resource group is effective in the specific date.
- ResourceGroupDescription
- ResourceGroupDisplayName
- ResourceGroupId: The resource group ID.
- ResourceGroupName
- ResourceGroups: Map of resource groups that contains this resource group. The map can be empty.
- Resources: Map of resources and resource groups contained in this resource group. The map can be empty.
- Services: Map of services. The map can be empty.

Remarks

Each entry in each internal map contains the object name in the key and in the value.

This method fails if the resource group and/or the change set do not exist.

GetResourceGroupDetailsByDate

Description

The method returns a map with the resource group details.

This method returns the resource group details true to the given date. If the third parameter is false it contains uncommitted changes as well. The returned map is the resource group details map as given by *GetResourceGroupDetails* (see page 100).

Syntax

GetResourceGroupDetailsByDate (ResourceGroupName, EffectiveDate, CommittedOnly)

Parameters

Name	Type	Description
ResourceGroupName	string	
EffectiveDate	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.
CommittedOnly	boolean	This parameter is optional. The default is **False**. When **True**, only committed values are retrieved. When **False**, get value and consider uncommitted values.

Return Value

See *GetResourceGroupDetails* (see page 100).

Remarks

This method fails if the resource group does not exist.

GetResourceGroupGeneralDetails

Description

This method returns a map with the resource group general details.

Syntax

GetResourceGroupGeneralDetails (ResourceGroupName)

Parameters

Name	Type	Description
ResourceGroupName	string	The name of the resource group.

Return Value

A map containing the following entries:

- ResourceGroupDescription
- ResourceGroupDisplayName
- ResourceGroupId: The Resource Group ID.
- ResourceGroupName

The general details are not version controlled.

Remarks

This method fails if the resource group does not exist.

IsResourceGroupExistsInTime

Description

This method checks whether the resource group committed as effective is in the given timeframe.

Syntax

IsResourceGroupExistsInTime (ResourceGroupName, EffectiveDate)

Parameters

Name	Type	Description
ResourceGroupName	string	

Name	Type	Description
EffectiveDate	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.

Return Value

True if the resource group committed as effective is in the given timeframe, **False** if it is not.

Remarks

This method fails if the resource group does not exist.

IsResourceGroupExists

Description

This method checks whether the specified resource group exists.

Syntax

IsResourceGroupExists (ResourceGroupName)

Parameter(s)

Name	Type	Description
ResourceGroupName	string	The name of the resource group.

Return Value

True if the resource group exists, **False** if it does not.

Remarks

None.

IsResourceGroupExistsInResourceVersion

Description

This method checks whether the resource group committed as effective is in the timeframe from the given change set.

Syntax

IsResourceGroupExistsInResourceVersion (ResourceGroupName, ChangeSetName)

Parameters

Name	Type	Description
ResourceGroupName	string	The name of the resource group.
ChangeSetName	string	The name of the change set.

Return Value

True if the resource group committed as effective is in the timeframe from the given change set, **False** if it is not.

Remarks

This method fails if the resource group and/or the change set do not exist.

SearchResourceGroups

Description

This method searches resource groups that are committed as effective in the given date.

Syntax

SearchResourceGroups (EffectiveDate, ResourceGroupNamePattern)

Parameters

Name	Type	Description
EffectiveDate	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.
ResourceGroupNamePattern	string	Can use wildcards.

Return Value

A map of resource groups, where the key and the value of each entry are both ResourceGroupName.

Remarks

This method fails if the resource group does not exist.

Parameters are optional.

The default is an empty string, which means to ignore the parameter.

If the effective date is empty, every resource group that is committed is returned as effective at some point in time.

UpdateResourceGroupByMap

Description

This method updates a resource group details on the effective date defined within the change set.

Syntax

UpdateResourceGroupByMap (ResourceGroupName,ResourceGroupDetails)

Parameters

Name	Type	Description
ResourceGroupName	string	The name of the resource group.
ResourceGroupDetails	Map	Map of resource group details is in the structure of the map returned from GetResourceGroupDetails method. The ResourceGroups, Resources, and Services internal maps can be empty. The ResourceGroupId entry is not relevant. The ResourceTypes entry must contain at least one resource type. The CustomAttributes internal map must contain all of the custom attributes that are attached to the resource groups.

Remarks

We recommend calling "Tools.UpdateResourceGroupByMap" (or "Tools.UpdateResourceByMap") instead of multiple calls to "Tools.UpdateResourcesCustomAttribute".

Example

```
Option Explicit

Const CHANGESET_NAME = "Default"

'This script assumes that the following resource Group exists
Const RESOURCE_TEMPLATE_NAME = "ResourceGroupTemplate"

Sub OnLoad()

  'Tools.log "Integration Script : In OnLoad procedure", "I"
End Sub

Sub Main()

  Dim resourceGroupName
  resourceGroupName = "MyResourceGroup1"

  Dim aMap
  Set aMap = Tools.GetResourceGroupDetails (RESOURCE_TEMPLATE_NAME, CHANGESET_NAME)

  aMap("ResourceName") = resourceGroupName
  'aMap("ChangeSetName") = CHANGESET_NAME
  aMap("CustomAttributes")("Owner")("CustomAttributeValue") = "Joe"
  aMap("CustomAttributes")("SerialNumber")("CustomAttributeValue") = "123456"

  If Not Tools.IsResourceExists(resourceName) Then
            Tools.AddResourceGroupByMap(aMap)
      Else
            Tools.UpdateResourceGroupByMap resourceGroupName, aMap
  End If

  'Tools.CommitResources resourceGroupName
  Tools.Commit

End Sub

Function Result
  'Tools.log " Integration Script : In Result function", "I"
  'Result =

End Function
```

UpdateResourceGroupGeneralDetails

Description

This method updates the resource general details. This update does not need to be committed by the *CommitResources* or *CommitChangeSets* methods.

Syntax

UpdateResourceGroupGeneralDetails (OldResourceGroupName, ResourceGroupGeneralDetails)

Parameters

Name	Type	Description
OldResourceGroupName	string	The name of the resource group before the update.
ResourceGroupGeneralDetails	Map	Map with the new general details. as returned from GetResourceGroupGeneralDetails (see page 103). The user can change any field in the map.

Remarks

This method fails if the resource group does not exist.

Resource Type Methods

AddResourceType

Description

This method adds a resource type.

Syntax

AddResourceType (ResourceTypeName, ResourceTypeDescription, EventTypes)

Parameters

Name	Type	Description
ResourceTypeName	string	Name of the resource type.
ResourceTypeDescription	string	Description of the resource type.

Name	Type	Description
EventTypes	string/map/null	Name of an event type or map of event types (can be empty).

Return Value

The new resource type ID.

Remarks

None.

GetResourceTypeDetails

Description

This method returns a map with the resource type details.

Syntax

GetResourceTypeDetails (ResourceTypeName)

Parameters

Name	Type	Description
ResourceTypeName	string	The name of the resource type.

Return Value

A map containing the following entries:

- EventTypes: Map of event types.
- ResourceTypeDescription
- ResourceTypeName

Remarks

Each entry in each map contains the object name in the key and in the value.

This method fails if the resource type does not exist.

IsResourceTypeExists

Description

This method checks whether the specified resource type exists.

Syntax

IsResourceTypeExists (ResourceTypeName)

Parameters

Name	Type	Description
ResourceTypeName	string	The name of the resource type.

Return Value

True if the resource type exists, **False** if it does not.

Remarks

None.

Resource Version Methods

The Resource Version entity was replaced (in version 4.0) by the Change Set entity. Therefore, alternative methods to the Resource Version methods were added. Where applicable, in each Resource Version method, the alternative Change Set method is indicated.

AddResourceVersion

Description

This method adds a new change set.

Syntax

AddResourceVersion (Name, EffectiveDate, Description)

Parameters

Name	Type	Description
Name	string	The name of the change set.
EffectiveDate	string	Represents date in the script's time zone. It can contain hours, minutes, and seconds.

Name	Type	Description
Description	string	

Remarks

The default date is taken from the EffectiveDate parameter.

The alternative Change Set method is *AddChangeSet* (see page 49).

IsResourceVersionExists

Description

This method checks whether a change set with the given name exists.

Syntax

IsResourceVersionExists (ChangeSetName)

Parameters

Name	Type	Description
ChangeSetName	string	The name of the change set.

Return Value

True if a change set with the given name exists, **False** if it does not.

Remarks

The alternative Change Set method is *IsChangeSetExists* (see page 50).

IsPreliminaryResourceVersion

Description

This method checks whether a change set with the given name exists.

Syntax

IsPreliminaryResourceVersion (ChangeSetName)

Parameters

Name	Type	Description
ChangeSetName	string	The name of the change set.

Return Value

True if a change set with the given name exists (even if the change set does not contain uncommitted changes), **False** if it does not.

Remarks

This method always returns **True** if a change set with the given name exists, even if the change set does not contain uncommitted changes.

This method fails if the change set does not exist.

SearchResourceVersions

Description

This method returns a map of change sets ordered by the effective time.

Syntax

SearchResourceVersions (Status, EffectiveDate, NamePattern)

Parameters

Name	Type	Description
Status	string	The Change Set status. Legal values are: PRELIMINARY, ACTIVE, or an empty string.
EffectiveDate	string	Represents date in the script's time zone, it can contain hours, minutes, and seconds.
NamePattern	string	Can use wildcards.

Return Values

A map, where each entry contains the following fields:

- ResourceVersionDescription (string)
- ResourceVersionEffectiveDate
- ResourceVersionName (string)
- ResourceVersionStatus (string): Always contains PRELIMINARY.

Remarks

All parameters are strings.

Parameters are optional.

The default values are empty strings. An empty string means that the parameter does not participate in the search.

The search ignores the Status parameter.

The EffectiveDate parameter is checked against the change set's default date.

The alternative Change Set method is *SearchChangeSets* (see page 51).

Role Methods

IsRoleExists

Description

This method checks whether the specified role exists.

Syntax

IsRoleExists (RoleName)

Parameters

Name	Type	Description
RoleName	string	The name of the role.

Return Value

True if the role exists, **False** if it does not.

Remarks

None.

SearchRoles

Description

This method returns a map of roles.

Syntax

SearchRoles (RoleNamePattern, RoleDescriptionPattern)

Parameters

Name	Type	Description
RoleNamePattern	string	Can use wildcards.
RoleDescriptionPattern	string	Can use wildcards.

Return Value

A map of roles.

Remarks

Role name is used as the entry key and entry value.

Service Methods

AddService

Description

This method adds a new service and returns the new service ID.

Syntax

AddService (ServiceName, ServiceDescription, ServiceFee, ServiceGroup)

Parameters

Name	Type	Description
ServiceName	string	The name of the service.
ServiceDescription	string	A description of the service.
ServiceFee	string	Contains empty string or legal numeric positive value.

Name	Type	Description
ServiceGroups	string/map/null	Name of a service group or map of service groups (can be empty).

Return Value

The new service ID.

Remarks

None.

GetServiceDetails

Description

This method returns a map with the service details.

Syntax

GetServiceDetails (ServiceName)

Parameters

Name	Type	Description
ServiceName	string	The name of the service.

Return Value

A map containing the following entries:

- ServiceName
- ServiceDescription
- ServiceFee
- ServiceGroups – Map of service groups.

Remarks

Each entry in each service group contains the object name in the key and in the value.

This method fails if the service does not exist.

IsServiceExists

Description

This method checks whether the specified service exists.

Syntax

IsServiceExists (ServiceName)

Parameters

Name	Type	Description
ServiceName	string	The name of the service.

Return Value

True if the service exists, **False** if it does not.

Remarks

None.

SearchServices

Description

This method returns a map of services.

Syntax

SearchServices (ServiceNamePattern)

Parameters

Name	Type	Description
ServiceNamePattern	string	Can use wildcards.

Return Value

A map of services.

Remarks

Service name is used as the entry key and entry value.

Translation Entry Methods

All the Translation Entry methods get the translation entry ID as parameter.

DeleteEntry

Description

This method deletes the given entry.

Syntax

DeleteEntry (EntryId)

Parameters

Name	Type	Description
EntryId	integer	The translation entry ID.

Remarks

None.

IgnoreEntry

Description

This method updates the status of the given entry to Ignored.

Syntax

IgnoreEntry (EntryId)

Parameters

Name	Type	Description
EntryId	integer	The translation entry ID.

Remarks

None.

RestorePendingEntry

Description

This method updates the status of the given entry to Pending.

Syntax

RestorePendingEntry (EntryId)

Parameters

Name	Type	Description
EntryId	integer	The translation entry ID.

Remarks

None.

SetManualTranslationEntry

Description

This method sets the **UpdateManually** field in the translation entries table.

Syntax

SetManualTranslationEntry (EntryId, Value)

Parameters

Name	Type	Description
EntryId	integer	The translation entry ID.
Value	boolean	Values are **True** or **False**.

Remarks

None.

TranslationEntry

Description

This method translates the entry.

Syntax

TranslateEntry (EntryId, ObjectName)

Parameters

Name	Type	Description
EntryId	integer	The translation entry ID.
ObjectName	string	Can be resource name, event type name, service name, contract party name, time zone name, or another string (used for translation table with VALUE type).

Remarks

This method checks if the object exists in the appropriate table according to the translation table type. If the object exists, the translation entry is translated to this object.

User Methods

Most of the user methods get UserFullName as a parameter. UserFullName is userName@OrganizationName.

The user methods work only on non-internal users.

AddUserByMap

Description

This method adds a new user and returns the new user ID.

Syntax

AddUserByMap (UserDetails)

Parameters

Name	Type	Description
UserDetails	map	Map of user details. This map is the same as the map returned by GetUserDetails (see page 122).

Return Value

The new user ID.

Remarks

The map must contain the following entries:

- ContractParties - One of:
  - Null
  - One contract party
  - One contract party group
  - Map of contract parties and/or contract parties groups
- Roles - One of:
  - Null
  - One role
  - Map of roles
- UserDefaultPage - Legal values are:
  - ALERTS
  - CONTRACTS
  - FAVORITES
  - TAB_DASHBOARD
- UserDescription
- UserGroups - One of:
  - Null
  - One user group
  - Map of user groups
- UserName - Length 3 to 20 without @
- UserOrganization - Length 1 to 30 without @

- UserPassword - Length 3 to 10 without @
- UserRequirePasswordChange - Legal values are:
  - N
  - Y
- UserPasswordExpirationInterval - In days (1 to 366)

  Required when UserRequirePasswordChange=Y
- UserStatus - Legal values are:
  - ACTIVE
  - INACTIVE
- UserTimeZone

GetUserDetails

Description

This method returns a map with the user details. The map contains the following entries:

- ContractParties: Map of contract parties and Contract Parties Groups.
- Roles: Map of roles.
- UserDefaultPage
- UserDescription
- UserFullCustomersAccess: Y or N define whether the user has access to all the customers.
- UserFullName: Contains name and organization.
- UserGroups: Map of user groups.
- UserName
- UserOrganizationName
- UserPassword
- UserPasswordExpirationInterval: In days.
- UserRequirePasswordChange: Legal values are N and Y.
- UserStatus
- UserTimeZone

Syntax

GetUserDetails (UserFullName)

Parameters

Name	Type	Description
UserFullName	string	UserName@OrganizationName

Remarks

If the value is Yes, the UserGroups map contains all the contract party groups.

This map does not contain the Accessible Contract Parties.

The password contains an empty string.

GetUserFullName

Description

This method returns a string with the user full name (UserName@OrganizationName).

Syntax

GetUserFullName (UserName, OrganizationName)

Parameters

Name	Type	Description
UserName	string	The name of the user.
OrganizationName	string	The name of the organization.

Return Value

A string with the user full name (UserName@OrganizationName).

Remarks

This method concatenates the given strings.

This method does **not** check if the user exists or if the user name is legal.

GetUserName

Description

This method returns the user name from the user full name.

Syntax

GetUserName (UserFullName)

Parameters

Name	Type	Description
UserFullName	string	The user full name.

Return Value

The string from the beginning up to the first '@'. If the '@' character does not exist, it returns an empty string.

Remarks

This method does **not** check if the user exists or if the user full name is legal.

IsUserExists

Description

This method checks whether the specified user exists.

Syntax

IsUserExists (UserFullName)

Parameters

Name	Type	Description
UserFullName	string	The user full name.

Return Value

True if the user exists, **False** if it does not.

Remarks

None.

SearchUsers

Description

This method returns a map of users.

Syntax

SearchUsers (UserStatus, UserNamePattern, OrganizationNamePattern, UserDescriptionPattern)

Parameters

Name	Type	Description
UserStatus	string	The user status.

Name	Type	Description
UserNamePattern	string	
OrganizationNamePattern	string	
UserDescriptionPattern	string	

Return Value

A map of users, where user full name is the key entries and value entries in the map.

Remarks

All the parameters are optional.

The default is an empty string, which means to ignore the parameter.

UpdateUserByMap

Description

This method performs a user update. Using this method, you can change the user fields, attached user groups, attached contract party and contract party groups, and attached roles.

Syntax

UpdateUserByMap (UserFullName, UserDetails)

Parameters

Name	Type	Description
UserFullName	string	UserName@OrganizationName
UserDetails	map	Map of user details (same restrictions as AddUserByMap (see page 119)).

Remarks

The first parameter is the current user full name.

The user can change the name and or organization name by setting new values to the UserName and OrganizationName entries.

User Group Methods

AddUserGroupByMap

Description

This method adds a new user group and returns the new user group ID.

Syntax

AddUserGroupByMap (UserGroupDetails)

Parameters

Name	Type	Description
UserGroupDetails	map	Map of user group details. This map is the same as the map returned by GetUserGroupDetails (see page 127).

Remarks

The map must contain the following entries:

- ContractParties - One of:
  - Null
  - One contract party
  - One contract party group
  - Map of contract parties and/or contract parties groups
- Roles - One of:
  - Null
  - One role
  - Map of roles
- UserGroupDescription
- UserGroupName
- Users - One of:
  - Null
  - One user
  - Map of users

DeleteUserGroup

Description

This method deletes the given user group.

Syntax

DeleteUserGroup (UserGroupName)

Parameters

Name	Type	Description
UserGroupName	string	The name of the user group.

Remarks

None.

GetUserGroupDetails

Description

This method returns a map with the user group details.

Syntax

GetUserGroupDetails (UserGroupName)

Parameters

Name	Type	Description
UserGroupName	string	The name of the user group.

Return Value

A map containing the following entries:

- ContractParties: Map of contract parties and contract parties groups.
- Roles: Map of roles.
- UserGroupDescription
- UserGroupName
- Users: Map of users.

Remarks

None.

IsUserGroupExists

Description

This method checks whether the specified user group exists.

Syntax

IsUserGroupExists (UserGroupName)

Parameters

Name	Type	Description
UserGroupName	string	The name of the user group. Length is 1 to 50 characters.

Return Value

True if the user group exists, **False** if it does not.

Remarks

None.

SearchUserGroups

Description

This method returns a map of user groups.

Syntax

SearchUserGroups (UserGroupNamePattern, UserGroupDescriptionPattern)

Parameters

Name	Type	Description
UserGroupNamePattern	string	
UserGroupDescriptionPattern	string	

Return Value

A map of user groups, where user group name is the key and the value of each entry.

Remarks

All parameters are optional.

The default is an empty string, which means to ignore the parameter.

UpdateUserGroupByMap

Description

This method performs a user group update. Using this method, you can change the user group name, description, attached users, attached contract parties and contract party groups, and attached roles.

Syntax

UpdateUserGroupByMap (UserGroupName, UserGroupDetails)

Parameters

Name	Type	Description
UserGroupName	string	The name of the user group.
UserGroupDetails	map	A map of contract party details, as returned by GetUserGroupDetails (see page 127).

Remarks

The first parameter is the current user group name.

The user can change the name by setting a new name to the UserGroupName entry.

Translation Entry Fields

Adapters are translated in the translation table by using either a sting or a collection of strings. This is defined in advanced by the user.

The following table lists the Translation Entry fields that can be used in translation scripts.

- There can be up to five field names.
- There can be up to five field values (one for each field name), and their inputs are set to be translated as a group in the defined order.

Field	Type	Description
ENTRY_ID	NUMBER	Translation unique entry ID.
TABLE_ID	NUMBER	Translation unique table ID (the entry belongs to this table from the translation table definition).
TABLE_NAME	VARCHAR2(30)	Translation table name (from the translation table definition).
TABLE_TYPE	VARCHAR2(30)	Translation table type (from the translation table definition). Legal values: RESOURCE, EVENT_TYPE, VALUE, SERVICE, CONTRACT_PARTY, TIME_ZONE.
TABLE_RETURN_TYPE	VARCHAR2(10)	The translation returns a value with this type (from the translation table definition). Legal values: INTEGER, STRING, FLOAT.
FIELD_NAME_1	VARCHAR2(50)	Name of the first input field (from the translation table definition).
FIELD_NAME_2	VARCHAR2(50)	Name of the second input field (from the translation table definition).
FIELD_NAME_3	VARCHAR2(50)	Name of the third input field (from the translation table definition).
FIELD_NAME_4	VARCHAR2(50)	Name of the fourth input field (from the translation table definition).
FIELD_NAME_5	VARCHAR2(50)	Name of the fifth input field (from the translation table definition).

Field	Type	Description
FIELD_VALUE_1	VARCHAR2(256)	The value in the first field to be translated (from the translation entries).
FIELD_VALUE_2	VARCHAR2(256)	The value in the second field to be translated (from the translation entries).
FIELD_VALUE_3	VARCHAR2(256)	The value in the third field to be translated (from the translation entries).
FIELD_VALUE_4	VARCHAR2(256)	The value in the fourth field to be translated (from the translation entries).
FIELD_VALUE_5	VARCHAR2(256)	The value in the fifth field to be translated (from the translation entries).
STATUS	VARCHAR2(15)	The translation entry status. Legal values: PENDING, TRANSLATED, IGNORED.
MANUAL_ TRANSLATION	CHAR(1)	Whether the user will do the translation. Legal values: Y, N.
RESOURCE_ID	NUMBER	The resource ID that the entry was translated to. Only for entries from table with TABLE_TYPE='RESOURCE'.
RESOURCE_NAME	VARCHAR2(100)	The resource name that the entry was translated to. Only for entries from table with TABLE_TYPE='RESOURCE'.
EVENT_TYPE_ID	NUMBER	The event type ID that the entry was translated to. Only for entries from table with TABLE_TYPE='EVENT_TYPE'.
EVENT_TYPE_ NAME	VARCHAR2(30)	The event type name that the entry was translated to. Only for entries from table with TABLE_TYPE=’EVENT_TYPE’.
SERVICE_ID	NUMBER	The service ID that the entry was translated to. Only for entries from table with TABLE_TYPE=SERVICE'.
SERVICE_NAME	VARCHAR2(30)	The service name that the entry was translated to. Only for entries from table with TABLE_TYPE=’SERVICE’.
CONTRACT_ PARTY_ID	NUMBER	The service ID that the entry was translated to. Only for entries from table with TABLE_TYPE=’CONTRACT_PARTY’.
CONTRACT_ PARTY_NAME	VARCHAR2(60)	The service name that the entry was translated to. Only for entries from table with TABLE_TYPE=’CONTRACT_PARTY’.
TIMEZONE_ID	NUMBER	The time zone ID that the entry was translated to. Only for entries from table with TABLE_TYPE=’TIMEZONE’.

Field	Type	Description
TIMEZONE_NAME	VARCHAR2(30)	The time zone name the entry translated to. Only for entries from table with TABLE_TYPE='TIMEZONE'.
TRANSLATED_TO	VARCHAR2(100)	The translated value for entries from table with TABLE_TYPE='VALUE'.
CREATED_BY_ USER_ID	NUMBER	The ID of the user who created the entry. This field is null if the entry was created by an adapter.
CREATED_BY_ USER_NAME	VARCHAR2(30)	The name of the user who created the entry. This field is null if the entry was created by an adapter.
CREATED_BY_ ADAPTER_ID	NUMBER	The ID of the adapter that created the entry. This field is null if the entry was created manually.
CREATED_BY_ ADAPTER_NAME	VARCHAR2(30)	The name of the adapter that created the entry. This field is null if the entry was created manually.
LAST_ACTION_ DATE	DATE	The date that the entry was last activated manually.
CREATE_DATE	DATE	The date that the entry was created.
MODIFY_DATE	DATE	The date of the last time that the entry was updated.

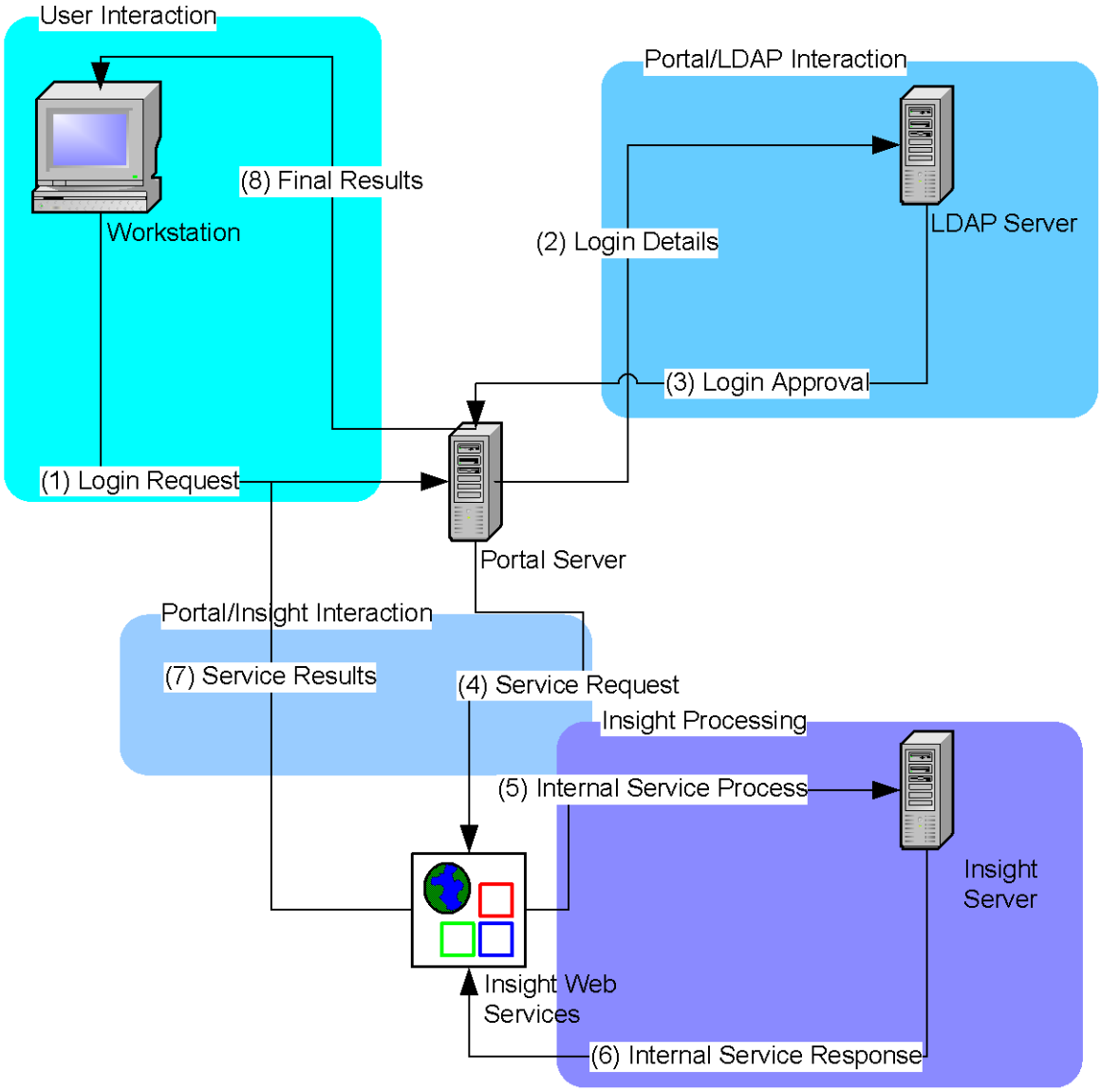
Chapter 7: Portal Integration

This section contains the following topics:

Overview

In a portal environment, the user logs from a workstation into the portal server application. The user's credentials are authenticated with the LDAP server and the CA Business Service Insight application. The user can now interact with the CA Business Service Insight Web Services (for example, getting a report list or displaying a specific report).

The following figure depicts the portal and SSO integration using CA Business Service Insight functions. For the available Web Services, see Web Services (see page 137).



Dashboard

The CA Business Service Insight Dashboard allows users to host dashviews in external sites. This is accomplished by calling the DashboardWebPart.aspx page of the Dashboard.

When a dashview is hosted in an external site it is loaded in view mode only. Users cannot modify the dashview or any widgets on it. Users do, however, have all the viewing capabilities of the dashview, as follows:

- Analyze
- Details
- Refresh
- Notes (can add new notes)
- View
- Navigate

To host dashviews in an external site:

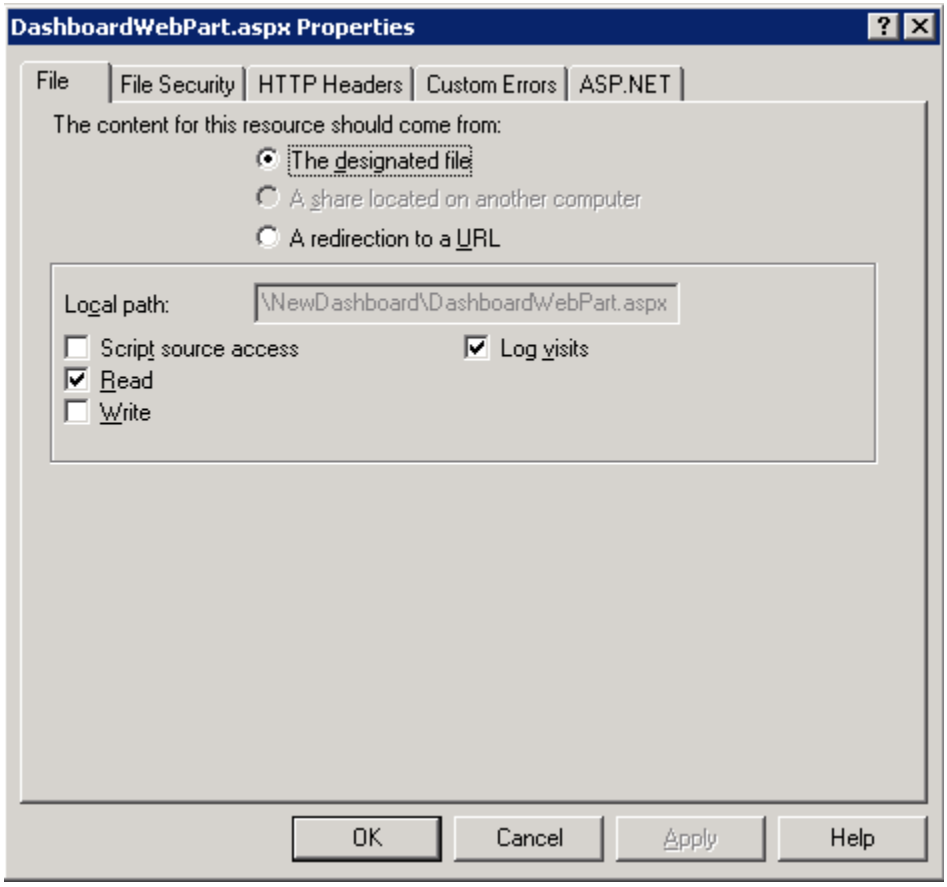
1. Call the DashboardWebPart.aspx page with the dashviewId, username, and organization parameters.
2. The URL should look like: http://[ServerName]/NewDashboard/DashboardWebPart.aspx?dashviewId=[]&username=[]&organization=[].

   Note: This page performs a silent login to CA Business Service Insight in order to let the user view the dashview page.

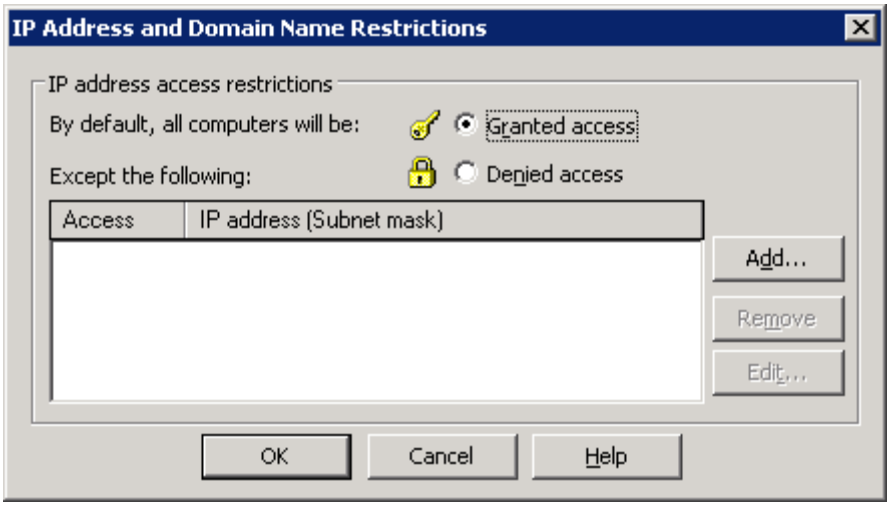
For added security, the dashview page can be configured to allow access only to specific IP address(es). This is performed through the Internet Information Services (IIS) Manager.

To configure dashview page to allow access only to specific IP address(es):

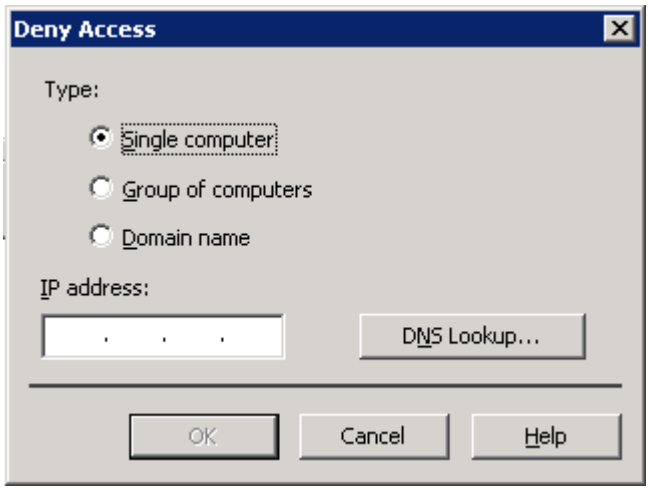
1. In *Control Panel*, double click on **Administrative Tools**. The *Administrative Tools* window is displayed.
2. Double-click on **Internet Information Services**. The *Internet Information Services (IIS) Manager* window is displayed.
3. Go to **<machine name> (local computer) > Web Sites > CA Cloud Insight > NewDashboard**.
4. Right click on **DashboardWebPart.aspx** and select **Properties**. The *DashboardWebPart.aspx Properties* window is displayed.



5. Click on the **File Security** tab.
6. In the *IP address and domain name restrictions* section, click **Edit**. The *IP Address and Domain Name Restrictions* window is displayed.



7. Select **Denied access**.
8. Click **Add**. The *Deny Access* window is displayed.



9. Grant access to the desired IP address and click **OK**.

Web Services

The CA Business Service Insight web services platform is based on a Services Oriented Architecture (SOA) using standards such as XML, SOAP and WSDL. Web services enable different languages and disparate platforms to communicate with each other and share data using remote connectivity. Insight Authentication is mandatory when interfacing with web services.

CA Business Service Insight provides web services for contracts, reports and authentication:

- WebServices/Contracts.asmx
- WebServices/InsightAuth.asmx
- WebServices/Reports.asmx

Web services are located in: <Insight Web>\WebServices.

Methods for Web Services

Web Services methods are divided into three groups: Report, Contract, and Authentication methods. The following three tables list the appropriate methods for each method group.

Report Methods	
GetAllReports (see page 139)	GetReportsByFolderID (see page 148)
GetAllReportsAdvanced (see page 141)	GetReportsByFolderIDAdvanced (see page 150)
GetFolderList (see page 140)	GetReportsByFolderName (see page 152)
GetMyReports (see page 144)	GetReportsByFolderNameAdvanced (see page 154)
GetMyReportsAdvanced (see page 145)	GetReportData (see page 157)

Contract Methods	
GetContract (see page 162)	GetContractAdvanced (see page 163)

Authentication Methods	
AuthenticateUser (see page 165)	ClearSessionContext (see page 166)

Remember, before using a Report or Contract method, a parameter called sessionID must be retrieved, as shown in the following procedure.

To retrieve the sessionID parameter:

1. Use the AuthenticateUser method to retrieve a token.
2. Perform a Silent Login using this token.
3. Retrieve the sessionID.
4. Use a Report or Contract method with the sessionID.

For more information see Single Sign-On (SSO) (see page 11).

Report Methods

GetAllReports

Description

This method returns full content of all reports in all folders, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetAllReports (**`string` **sessionID)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

GetFolderList

Description

This method returns a list of all folders that exist in CA Business Service Insight.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetFolderList (**`string` **sessionID)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.

Field Name	Type	Description
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
is_my_folder	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

GetAllReportsAdvanced

Description

This method returns full content of all reports in all folders, according to specified search criteria, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetAllReportsAdvanced (**`string` **sessionID,** `string` **foldername,** `string` **CriteriaXML)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
foldername	string	Contains folder name.

CriteriaXML	string	Contains search criteria.
		`<Criteria>`
		`<Name></Name>`
		`<ReportTypes></ReportTypes>`
		`</Criteria>`
		CriteriaXML parameter contains search criteria for report selection:
		Name node contains mask for searching report by name. Asterisks (*) symbols are allowed for a mask search.
		ReportTypes node contains a comma-separated list of report types to search. Available report types include FreeForm, Group, Compound, and Normal. Type names are case-insensitive.
		Example:
		`<Criteria>`
		`<Name>ServiceRep*</Name>`
		`<ReportTypes>FreeForm,Group</ReportTypes>`
		`</Criteria>`
		When criteriaXML parameter is an empty string, all report types are included in the resulting list.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.

Field Name	Type	Description
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.

Request	Result
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”.
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetMyReports

Description

This method returns full content of My Reports folder, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetMyReports (**`string` **sessionID)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).

Field Name	Type	Description
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

GetMyReportsAdvanced

Description

This method returns all reports that are located in My Reports folder, according to specified search criteria, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetMyReportsAdvanced (**`string` **sessionID,** `string` **CriteriaXML)**

Parameter

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.

Name	Type	Description

CriteriaXML	string	Contains search criteria.
		`<Criteria>`
		`<Name></Name>`
		`<ReportTypes></ReportTypes>`
		`</Criteria>`
		CriteriaXML parameter contains search criteria for report selection:
		Name node contains mask for searching report by name. Asterisks (*) symbols are allowed for a mask search.
		ReportTypes node contains a comma-separated list of report types to search. Available report types include FreeForm,Group, Compound, and Normal. Type names are case-insensitive.
		Example:
		`<Criteria>`
		`<Name>Service Rep*</Name>`
		`<ReportTypes>FreeForm,Group</ReportTypes>`
		`</Criteria>`
		When criteriaXML parameter is an empty string, all report types are included in the resulting list.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).

Field Name	Type	Description
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”.
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetReportsByFolderID

Description

This method returns full content of a specified folder, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetReportsByFolderID (**`string` **sessionID,** `int` **folderID)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
folderID	int	Contains folder ID.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”.
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetReportsByFolderIDAdvanced

Description

This method returns full content of a specified folder, according to specified criteria, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetReportsByFolderIDAdvanced (**`string` **sessionID,** `int` **folderID,** `string` **CriteriaXML)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
folderID	int	Contains folder ID.

CriteriaXML	string	Contains criteria XML.
		`<Criteria>`
		`<Name></Name>`
		`<ReportTypes></ReportTypes>`
		`</Criteria>`
		CriteriaXML parameter contains search criteria for report selection:
		Name node contains mask for searching report by name. Asterisks (*) symbols are allowed for a mask search.
		ReportTypes node contains comma-separated list of report types to search. Available report types are Freeform, Group, Compound, and Normal. Type names are case-insensitive.
		Example:
		`<Criteria>`
		`<Name>Service Rep*</Name>`
		`<ReportTypes>FreeForm,Group</ReportTypes>`
		`</Criteria>`

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetReportsByFolderName

Description

This method returns full content of a specified folder, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetReportsByFolderName (**`string` **sessionID,** `string` **folder)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
foldername	string	Contains folder name.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”.
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetReportsByFolderNameAdvanced

Description

This method returns full content of a specified folder according to specified search criteria, excluding booklets and shortcuts.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetReportsByFolderNameAdvanced (**`string` **sessionID,** `string` **foldername,** `string` **CriteriaXML)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
foldername	string	Contains folder name.

Name	Type	Description

CriteriaXML	string	Contains search criteria.
		`<Criteria>`
		`<Name></Name>`
		`<ReportTypes></ReportTypes>`
		`</Criteria>`
		CriteriaXML parameter contains search criteria for report selection:
		Name node contains mask for searching report by name. Asterisks (*) symbols are allowed for a mask search.
		ReportTypes node contains a comma-separated list of report types to search. Available report types are Freeform, Group, Compound, and Normal. Type names are case-insensitive.
		Example:
		`<Criteria>`
		`<Name>ServiceRep*</Name>`
		`<ReportTypes>FreeForm,Group</ReportTypes>`
		`</Criteria>`
		When criteriaXML parameter is an empty string, all report types are included in the resulting list.

Return Value

Returns a dataset in the following format:

Table name: tblreports

Field Name	Type	Description
item_id	Long	ID to perform authentication check.
item_name	String	Name of the report.
user_id	Long	ID of User that owns the report.
folder_id	Long	ID of the folder that the report is located in.
folder_name	String	Name of the folder that the report is located in.
item_description	String	Report description.
item_type	String	Item type (report).
report_type	String	Report type (normal, compound, group, booklet, free-form).

Field Name	Type	Description
is_executable	boolean	Shows if report is executable.
modify_date	Date	Last modify date.
schedule_id	Long	ID of schedule.

Errors

If any errors occur, web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remarks

All parameters are mandatory.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Call method without permission to view reports	Throws error NO_USER_PERMISSION.
Call method only with SessionID	Default filter data is applied (All reports types in my reports).
FolderID = 0	Returns all reports that exist in root folder only.
FolderID = 0 Subfolders = 1	Returns all reports in all folders.
FolderID = 0 FolderName = “FolderName"	Returns all reports that exist in root folder only. Folder name is not relevant.
FolderName = “FolderName”	Returns all reports that exist in folder by name “FolderName”.
Specify invalid FolderID or FolderName	Throws error OE_FOLDER_NOT_EXIST.

GetReportData

Description

This method returns report data as XML and optionally generates a GIF image with the report chart.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`XmlNode` **GetReportData (**`string` **sessionID,** `int` **FavoriteID,** `int` **pictureRequest,** `int` **width,** `int` **height)**

Parameter

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
favoriteID	int	ID of the report that needs to be retrieved.
pictureRequest	int	Generate report picture and return path. 1 = yes; 0 = no.
width	int	Width of picture.
height	int	Height of picture.

Return Value

Report outputs differ according to report type. The Reports Web service adds a path specifies path to the generated report image:

For normal, compound and free-form reports (with chart):

```
<RESULT ... HAS_CHART ="1/0">
    <ITEM ...>
        .
        .
        .
      <PICTURE_PATH>"<REPORTID><NUM><COUNT>".JPG </ PICTURE_PATH >
     </ITEM>
</RESULT>
```

For Group reports:

```
<RESULT ... HAS_CHART =”1/0”>
    <ITEM ...>
        .
        .
        .
      <PICTURE_PATH>"<REPORTID><NUM><COUNT>".JPG </ PICTURE_PATH >
     </ITEM>
     <ITEM ...>
        .
        .
        .
      <PICTURE_PATH>"<REPORTID><NUM><COUNT>".JPG</ PICTURE_PATH >
     </ITEM>

</RESULT>
```

Remarks

All parameters are mandatory.

If an image is not requested, only the report attributes are returned.

The parameters “PictureRequest”, “width”, and “height” are relevant only to reports that have pictures.

Free-form reports that contain parameters are not supported by this service.

In Report Outputs, HAS_CHART="1" indicates that for this report a report image can be generated with the new tag PICTURE_PATH. The location of the tag depends on the report type. The template "<REPORTID><NUM><COUNT>".JPG will be used to store the image name, where:

- NUM: Incremented number of generated reports per session.
- COUNT: Count of inner reports that are included in the reports.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Normal report	Report data is returned.
Normal report with chart	Report data is returned, report chart is generated and included in XML data.
Freeform report with chart	Report data is returned, report chart is generated and included in XML data.

Request	Result
Freeform report with parameters	While report parameters have valid default values that report data is returned, otherwise error is returned.
Freeform report without chart	Report data is returned.
Compound report	Report data is returned.
Group report	Report data is returned.
Compound report with chart	Report data is returned, report chart is generated and included in XML data.
Group report with chart	Report data is returned, report chart is generated and included in XML data.
Report does not exist	Error is returned "OE_REPORT_NOT_EXIST".
Illegal report id is supported	Error is returned "OE_ILLEGAL_PARAMETER".
Report is not permitted	Error is returned "OE_NO_USER_PERMISSION".

GetReportDataAndExport

Description

This method returns report data XML and exports the report.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`XmlNode` **GetReportDataAndExport (**`string` **sessionID,** `int` **favoriteID, string parametersXML,** `string` **exportXML,** `string` **criteriaXML)**

Parameter

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.
favoriteID	int	ID of the report that needs to be retrieved.
parametersXML	int	Relevant only for free-form reports. Optional.
exportXML	string XML	Defines the export parameters.
criteriaXML	string XML	Defines the method's behavior. Optional.

Return Value

This XML also contains information about exported results (file paths) according to following format:

```
<RESULT …>
        :
        :
<EXPORT_RESULTS>
<FILE_PATH>sessiondata/673994876/FF_1_Param_6.pdf</FILE_PATH>
</EXPORT_RESULTS>
</RESULT>
parametersXML - should be defined in “brief” format of parameters FreeForm official
XML. Values supplied by this XML are merged to FreeForm definition. Any irrelevant
section or parameter might be omitted.
<connection>
        <params>
                <param name='@UID'><value>user id</value></param>
                <param name='@PWD' ><value>password</value></param>
        </params>
</connection>
<query>
        <params>
                <param name='@PRM1'><value>value 1</value></param>
                <param name='@PRM2'><value>value 2</value></param>
        </params>
</query>
```

exportXML

The parameter exportXML should be defined as follows. Note that upper case is *mandatory*.

```
<REPORT_EXPORT>
        <EXPORT_TYPE>{PDF(default)|CHART|HTML|MHT|CSV|CSVDO}</EXPORT_TYPE>
                <INCLUDED>
<REPORT>
{CHART(default)| DATA | FILTERS | RAW_DATA |BRC | CORRECTIONS | EXCEPTIONS |
PENALTY_ADJ | PENALTY_ERR}
</REPORT>
<REPORT></REPORT>
                </INCLUDED>
</REPORT_EXPORT>
```

- The EXPORT_TYPE tag should be defined only once.
- The REPORT tag in the INCLUDED section represents the part of the report that will be exported. Multiple report parts can be exported by time.

criteriaXML

The parameter criteriaXML should be defined as follows. Note that tag names are *case sensitive*.

```
<Criteria>
<UseCache>{1(default) | 0}</UseCache>
<GenerateIfNoCached>{1(default) | 0}</GenerateIfNoCached>
<Width>400</Width>
<Height>200</Height>
</Criteria>
```

- A UseCache value of "1" means that previously cached report will be returned, even if current format does not correspond to the previously cached format.
- A GenerateIfNoCached value of "1" means to perform an export routine if the cached export results are not present and UseCache=1 or is omitted.

  Note: Caching generated (exported) report should be performed by API consumer. Files included in EXPORT_RESULTS in returned XML should be copied to the cache directory (registry key Insight\paths\ReportsXMLcache). Also, the result XML should be stored in the same location as follows: [repID].xml (for example, 1000.xml).

Errors

If any errors occur, the web service will log and throw the InsightWebServiceException error.

Error Name	Description
AUTH_ERROR	Authentication failed.
NO_USER_PERMISSION	User does not have permission to view reports.
FOLDER_NOT_EXIST	Specified folder does not exist.
OE_ILLEGAL_PARAMETER	Specified parameter is not valid.

Remark

The parameters parametersXML and criteriaXML are optional. A free-form report that contains parameters but parametersXML is empty, results in a report error according to the default FreeForm parameters section.

All parameters are passed as type string, to ensure flexibility in calling the method.

Sample Scenarios

Possible scenarios to generate request report data:

Request	Result
Normal report	Report data is returned.
Normal report with chart	Report data is returned, report chart is generated and included in XML data.
Freeform report with chart	Report data is returned, report chart is generated and included in XML data.
Freeform report with parameters.	While report parameters have valid default values that report data is returned, otherwise error is returned.
Freeform report without chart	Report data is returned.
Compound report	Report data is returned.
Group report	Report data is returned.
Compound report with chart	Report data is returned, report chart is generated and included in XML data.
Group report with chart	Report data is returned, report chart is generated and included in XML data.

Contract Methods

GetContract

Description

Returns a list of SLAs for the current user.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

EntityListData **GetContracts** (string EntityID);

Parameters

Name	Type	Description
EntityID	long	The ID of the entity.
EntityName	string	The name of the entity.

Return Value

The contract name and IDs, as type EntityListData.

Remarks

The EntityListData object is returned as a collection of EntityListItemData items, containing the name and ID of the contract/s.

GetContractAdvanced

Description

Returns a list of SLAs for the current user according to specified search criteria.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`DataSet` **GetContractsAdvanced (**`string` **sessionID,** `string` **criteria XML)**

Parameters

Name	Type	Description
sessionID	string	Current session ID to perform authentication check.

criteriaXML		criteriaXML parameter should be in the following format:
		`<Criteria>`
		`<Name>{search mask}</Name>`
		`<Status>{PRELIMINARY,PENDING,EFFECTIVE}</Status>`
		`</Criteria>`
		If criteriaXML is empty string, then criteria is not applied.

Return Value

Returns a dataset with the following fields (Tag Name and Text):

Tag Name/Text	Description
SLA_ID	ID of contract.
SLA_NAME	Name of contract.
LOCALE_ID	ID of locale.
LOCALE_NAME	Name of locale (for example, GMT).

Tag Name/Text	Description
CUSTOMER_ID	ID of customer.
CUSTOMER_NAME	Name of customer.
CUSTOMER_TYPE_NAME	Name of customer type.
CUSTOMER_TYPE_DISPLAY_NAME	
SLA_MODIFY_DATE	Date the contract was modified.
CURRENT_VER_ID	ID of the current version.
CURRENT_VER_STATUS	The status of the current version.
LATEST_VER_ID	ID of the latest version.
LATEST_VER_STATUS	Status of the last version of the contract.
SLA_STATUS_DISPLAY	
CUR_VER_STATUS_DISPLAY	
LATEST_VER_STATUS_DISPLAY	
SLA_STATUS	Status of the contract.
SLA_VERSIONS	Number of contract versions.
IS_PSL_PURGED	
SLA_DECODE_STATUS	
CUR_VER_NONCOMPILED_PNLT_COUNT	
CUR_VER_PENALTY_COUNT	
CUR_VER_RULE_COUNT	
CUR_VER_APPLICATION_COUNT	
CUR_VER_NONCOMP_SLALOM_COUNT	
CUR_VER_RT_ERROR_RULE_COUNT	
CUR_VER_HAS_COLLIDING_RULES	Whether the version has colliding rules.
WORKFLOW_STATUS	The workflow status.
SLA_VALID_ FROM	Date from which the contract is valid.
SLA_VALID_TO	Date when the contract is valid until.
SCORE	

Remarks

SessionID is mandatory.

Authentication Methods

AuthenticateUser

Description

Checks if user with such credentials (username and organization) exists and is active in CA Business Service Insight database.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`string` **AuthenticateUser (**`string` **username,** `string` **organization)**

Parameters

Name	Type	Description
userName	string	Contains user name.
organizationName	string	Contains organization name.

Return Value

Returns special encrypted token containing user ID and issue timestamp. If no such active user exists, null token is returned.

Remarks

None.

ClearSessionContext

Description

Deletes the content of the specified SessionData folder.

Syntax

A typical syntax for this method is shown. Note that the exact syntax to use depends on the type of protocol used to communicate with the web service.

`void` **ClearSessionContext (**`string` **sessionContextID)**

Parameter

Name	Type	Description
sessionContextID	string	Contains session context ID.

Return value

No return value.

Remarks

None.

Web Services Security

All the Web services, excluding Check Authorization, require CA Business Service Insight Session ID as the authentication base. If an invalid or non-existent Session ID is passed, no data is retrieved.

In addition, for security reasons, **WebServices** directory is configured to allowed access only from the portal IPs.

Web Parts

<o> supports third-party corporate portal solutions through a service-oriented architecture (SOA) implementation. This encompasses the delivery of standard Web Services and “out-of-the-box” available Web Parts for Microsoft SharePoint Services.

CA Business Service Insight can be integrated into corporate portal solutions using SOA-based web services.

The following capabilities can be achieved using CA Business Service Insight in a portal:

- Access to static and dynamic reports. Dynamic reports allow drill-down for root cause analysis.
- Parameterized free-form reports can be created and integrated into a portal, significantly reducing the time and effort for creating standardized free-form reports running over hundreds to thousands of data elements as parameters.
- The list of contracts modeled within CA Business Service Insight can be integrated into a portal, providing the user with the capability to view all accessible contracts and select one to review the contract details as an RTF file.
- Using <o>’s selection of Web Parts for Microsoft SharePoint Services, users can “drag and drop” reports from a list into an optimized viewing area.
- CA Business Service Insight can be integrated LDAP to provide single sign-on (SSO) capabilities and role-based access to CA Business Service Insight Web Services and Web Parts.

Installing Web Parts

Prior to installation, verify that Microsoft Windows SharePoint Services (SPS) is installed on your machine.

You are not required to install CA Business Service Insight and SPS on the same server. However, if this is your desired topology, you must create two separate websites. *Do not change the default website.* If you installed SPS prior to installing CA Business Service Insight, set it to a different website.

Copying files

The CA Business Service Insight web part DLL and DWP files must be copied to the SharePoint root folder bin (for example, C:\Inetpub\wwwroot\bin).

If this folder does not exist, create it and copy into it the following files from C:\Insight\Insight_8_0\Setup\WebParts:

- ContractList.dwp
- InsightWebParts.dll

- Report.dwp
- ReportList.dwp

Modifying the Configuration Files

Prior to adding the web parts to SPS, you must modify the configuration files to suit your requirements by registering the Web Part as safe.

To register a web part as safe:

1. Using Notepad or any other text editor, open the web.config file located in **C:\Inetpub\wwwroot**.
2. Under the <SafeControls> section, add the following line:

   ```
   <SafeControl Assembly="InsightWebParts, Version=1.0.0.0, Culture=neutral, PublicKeyToken=06840085de116645" Namespace="Insight.WebParts" TypeName="*" Safe="True" />
   ```

3. Verify that PublicKeyToken is correct by copying the InsightWebParts.dll file to the C:\Windows\Assembly folder (which is the Global Assembly Cache (GAC) folder) then right click and select Properties. The PublicKeyToken value should be 06840085de116645. Remove the file from the GAC.
4. Under the <SecurityPolicy> section add the following line:

   ```
   <trustLevel name="WSS_Minimal_Custom" policyFile="C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\60\config\wss_minimaltrust_custom.config" />
   ```

5. Under the </httpModules> section add the following line:

   ```
   <trustLevel name="WSS_Minimal_Custom" originUrl="" />
   ```

6. Save and close the file.
7. Go to C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\60\config.
8. Copy the wss_minimaltrust.config file and rename to wss_minimaltrust_custom.config
9. Using Notepad or any other text editor, open the wss_minimaltrust_custom.config file.

10. Under the </CodeGroup> section add the following lines:

```
<CodeGroup class="UnionCodeGroup" version="1" PermissionSetName="FullTrust">
<IMembershipCondition version="1" class="StrongNameMembershipCondition"
Name="InsightWebParts"
PublicKeyBlob="00240000048000009400000006020000002400005253413100040000010001
000907069C1276E74B2BAD4DBE97B0B9CA8D72F17E9A9A98482BC2AF4DCD3A018E45FDB585665
852722B8A69B90FD1247EE475D658379E24A3EFCDEF2AFCBE11BF3FC6174C8D5E79C8D83666F2
49FEB376B19145CE28A1AD4165BCCF23FF27C3461F8791E5D74C7CDF6341121E6BDA74EA68347
455E14E8DBCD5824146713805AE"/>
</CodeGroup>
```

11. Save and close the file.
12. In Control Panel, double-click on **Administrative Tools** and on **Computer Management**.
13. Under Services and Applications, click on **Internet Information Services**.
14. Right-click on **Web Sites** and select **Properties**.
15. Select the **Home Directory** tab and denote the Application pool.
16. Click **Cancel**.
17. Select **Application Pools**.
18. Right-click on the application pool you denoted in step 15 and select **Recycle**.
19. Go to **Start** then **Run** and type **iisreset**.
20. Click **OK**.

Enabling Session State on Microsoft Windows SharePoint

The session state should be enabled in the web.config file.

To enable session state:

1. Using Notepad or any other text editor, open the web.config file.
2. Under the <httpModules> section, add the following line:

```
<add name="Session" type="System.Web.SessionState.SessionStateModule"/>
```

Note: If this section already exists, verify that it is not in comments.

3. Under the </httpModules> section, verify that the following line appears as indicated below.

```
<pages enableSessionState="true" enableViewState="true" enableViewStateMac="true"
validateRequest="false" />
```

Note: By default, the enableSessionState attribute is set to **false**. If it is **false**, change it to **true**.

Specifying Organization Name and CA Business Service Insight Website Root Path

To specify the CA Business Service Insight organization name (which is the same for all SharePoint users) and the CA Business Service Insight website root path (used as the default path, unless the SharePoint administrator changes it in the specific web part settings), the following settings should be added into SharePoint web.config file just before the </configuration> closing tag at the end of the file:

```
<appSettings>
   <add key="InsightOrganizationName" value="Organization_Name" />
   <add key="OGURL" value="http://path_to_oblicore_website_root" />
 </appSettings>
```

Note: If the AppSettings section already exists in the web.config file, just add the key - do **not** duplicate the AppSettings section. If you have several SharePoint sites, repeat for each of the sites.

Adding Web Parts to the SharePoint Server Pages

CA Insight web parts can be dragged and dropped onto the SharePoint page. In order to drag and drop the web part, you first need to import the .dwp file.

To display a Web Part in a SharePoint Web Page:

1. Open your SharePoint Web site and navigate to the page where you want to add the Web Part. In the upper-right corner, from the Modify My Page (or Modify Shared Page if working in shared view) tab, click the arrow and select Add Web Parts, and then select Import.
2. Browse for the .dwp that you want to import in the bin folder (Report.dwp, ReportList.dwp, ContractList.dwp).
3. Click Upload.
4. Drag-and-drop the web part to the desired place in the page.
5. In the properties panel specify the root path of the CA Insight website (Miscellaneous/CA Cloud Insight URL) setting and other relevant settings. For more information, see Contract List Web Part Properties (see page 171) and Report Web Part Properties (see page 172).
6. Set up a connection between the web parts, if needed (for example, between ReportList and Report web part), in the web part settings context menu.

Contract List Web Part Properties

The Contract List web part displays a list of contracts titles, according to the current user's permissions. Clicking on the contract title opens a dialog with contract export, which enables opening exported contracts in MS Word.

In order to display a contract list, the web part calls the Contract List web service from the CA Business Service Insight website.

Once retrieved, the contract list is cached.

Note: The cache is invalidated every 10 minutes.

Insight URL

Alias	CA Business Service Insight URL
Description	Specifies path to CA Business Service Insight web services root folder.
Default	http://localhost
Example	http://Insight-server
Remarks	In SPS, CA Business Service Insight Web Part Properties, Miscellaneous.

NameMask

Alias	Search report mask
Description	Specifies search mask for report names.
Type	string
Default	None (no filter by report name)
Example	
Remarks	Asterisks are allowed.

Preliminary

Alias Show contracts with preliminary status.

Description Specifies whether or not contracts with preliminary status should be included in the list.

Type Boolean

Default True

Example

Remarks

Pending

Alias Show contracts with pending status.

Description Specifies whether or not contracts with pending status should be included in the list.

Type Boolean

Default True

Example

Remarks

Effective

Alias Show contracts with effective status.

Description Specifies whether or not contracts with effective status should be included in the list.

Type Boolean

Default True

Example

Remarks

Report Web Part Properties

The report web part opens a report page on the CA Business Service Insight website in a frame within the SharePoint page, displaying a single report chart.

The web part may be either static mode or dynamic mode.

- Static mode: The report chart is displayed as an image (jpg), without the possibility of drill-down.
- Dynamic mode: The report chart is displayed as an ActiveX object, allowing drill-down, and providing other report-related information, such as report data tables and report filter descriptions.

The web part may work in either autonomic mode or connected mode.

- Autonomic mode: The web part displays report by Report ID, provided in the Web Part properties panel during the web part configuration stage.
- Connected mode: The web part receives a Report ID from the ReportList web part when the user clicks on a report caption in the list of reports. To set the web part to connected mode, the SharePoint administrator must set the connection to the ReportList web part.

ReportID

Alias Report ID

Purpose Specifies ID of the report to display. This property appears in the Properties panel as drop-down list of available reports.

Default None

Example

Remarks

Insight URL

Alias CA Business Service Insight URL

Description Specifies path to CA Business Service Insight web services root folder.

Default http://localhost

Example http://Insight-server

Remarks In SPS, CA Business Service Insight Web Part Properties, Miscellaneous.

DisplayMode

Alias Display Mode

Purpose Specifies the report web part mode. Available values are Static and Dynamic.

Default Static

Example

Remarks

ShowFilter

Alias Show filter

Description Specifies whether or not the Filter tab should be displayed.

Default False

Remarks This property is relevant only when DisplayMode is Dynamic.

ReportWidth

Alias Report Width

Description Specifies width of report chart image, in pixels.

Default None

Example

Remarks This property is relevant only when DisplayMode is Static. If the property is not set, then the web part width is used (if specified).

ReportHeight

Alias Report Height

Description Specifies height of report chart image, in pixels.

Default None

Example

Remarks This property is relevant only when DisplayMode is Static. If the property is not set, then the web part height is used (if specified).

ReportList Web Parts Properties

The ReportList web part displays a list of report titles in the specified folder, according to the current user's permissions.

Note: The ReportList web part does not include booklets.

Clicking a report title causes reloading of the **Report** web part to display the specific report.

In order to display a report list, the web part calls the ReportList web service from the CA Business Service Insight website.

Note: To connect the ReportList web part to the Report web part, your SharePoint administrator must set up the connection to the Report web part.

Once retrieved, the report list is cached.

Note: The cache is invalidated every 10 minutes.

Insight URL

Alias CA Business Service Insight URL

Description Specifies path to CA Business Service Insight web services root folder.

Default http://localhost

Example http://Insight-server

Remarks In SPS, CA Business Service Insight Web Part Properties, Miscellaneous.

FolderID

Alias Reports Folder

Description Specifies folder in which to display the report list.

Type int

Default 0 (root folder)

Example

Remarks This property appears in the Properties panel as a drop-down list of available folders.

Normal

Alias Show normal reports

Description Specifies whether or not normal reports should be included in the list.

Type Boolean

Default True

Example

Remarks

Group

Alias Show group reports

Description Specifies whether or not group reports should be included in the list.

Type Boolean

Default True

Example

Remarks

Compound

Alias Show compound reports

Description Specifies whether or not compound reports should be included in the list.

Type Boolean

Default True

Example

Remarks

Freeform

Alias Show freeform reports

Description Specifies whether or not free-form reports should be included in the list.

Type Boolean

Default True

Example

Remarks

NameMask

Alias Search report mask

Description Specifies search mask for report names.

Type string

Default None (no filter by report name)

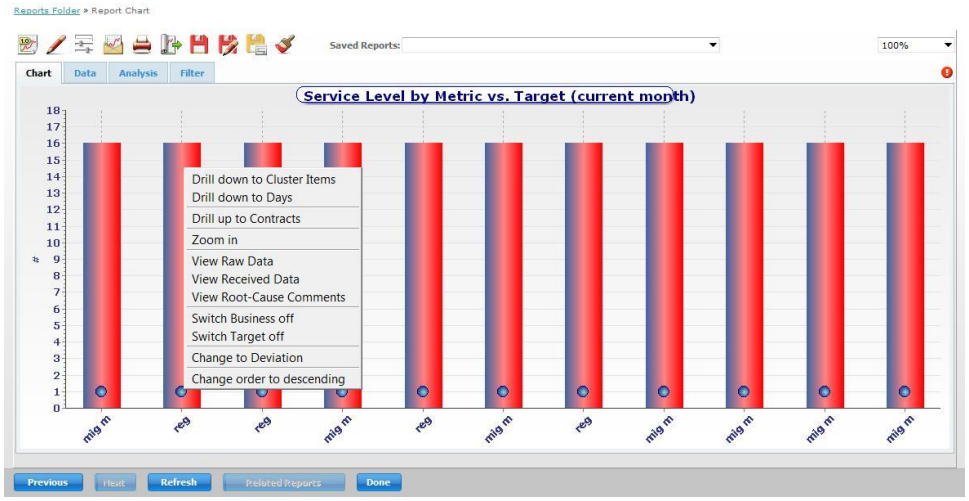
Example

Remarks Asterisks are allowed.

Web Part Example

Dynamic Report

This example illustrates a dynamic report accessed through a portal. By right-clicking on a chart in the portal, you can manipulate the report as desired.


Reports Folder » Report Chart
Saved Reports:
100%
Chart
Data
Analysis
Filter
Service Level by Metric vs. Target (current month)
Drill down to Cluster Items
Drill down to Days
Drill up to Contracts
Zoom in
View Raw Data
View Received Data
View Root-Cause Comments
Switch Business off
Switch Target off
Change to Deviation
Change order to descending
Previous
Next
Refresh
Related Reports
Done

Chapter 8: Open API

This section contains the following topics:

Overview

CA Business Service Insight's Open Interface provides connectivity with CA Business Service Insight, using standard secured web services.

The Open API exposes Insight's entities as complex data classes and exposes system functionality via web methods.

The Open Interface supports certificate-based data encryption, user authentication against Insight's user base, authorization for all methods invoked, and built-in data validation procedures. Data that is retrieved from the system through the Open Interface is filtered according to the predefined permissions of the user for whom credentials are provided. For more information, see API Infrastructure (see page 185).

The Open API consists of:

- Open API Interface Methods (see page 188)
- Open API Object Classes (see page 291)

API Backward Compatibility

Generally, CA Business Service Insight API backward compatibility is kept between one version and another. In CA Business Service Insight slight modifications were made to improve quality. Therefore, the API interface must be updated to take into account the changes made to the current version (for detailed information, see Changes to API (see page 180)).

To update the API to version 7.0 Service Pack 1:

1. Use the WSDL from <v> to create the Proxy classes for the Contract Service, Portfolio Service and Repository Service.
2. Recompile your API.

   If there are compilation errors:

   a. Locate the code that requires manual changes and update it according to the changes made to the new API (field order, type, etc.).

   b. Recompile your API again after making these changes.

Changes to API

Changes to the API were made to the:

- MetricData class
- DomainCategory methods
- API Interface

Changes to the MetricData Class

The following changes were made to the *MetricData* class:

- The *Target* field type was changed from long to double.
- A new Boolean field called *IsTargetLess* was added. The default value for the field is "false".

  Note: When the *IsTargetLess* field is "true", the value of Target is null, i.e., there is NO target.

The changes enabl the e *MetricData* class to have values of null (IsTargetLess = true) as well as zero and to have non-integer values.

Affected Methods

The nine methods affected by these changes are:

- CreateContractMetric
- CreateContractTemplateMetric
- CreateServiceDefinitionMetric
- GetContractMetricDetails
- GetContractTemplateMetricDetails
- GetServiceDefinitionMetricDetails
- UpdateContractMetricDetails
- UpdateContractTemplateMetricDetails
- UpdateServiceDefinitionMetricDetails

Changes to the DomainCategory Methods

The following changes were made to the *DomainCategory* methods:

- The *ID* and *ServiceDomain* fields were added to the *DomainCategoryData* classes and the internal order of the fields was modified slightly.
- The *ServiceDomain* field was removed from the *CreateDomainCategory* method.

- The *OptionalDoubleEntity* class was added to the API. The new structure contains the following two fields:

  Type Boolean; specifies whether or not a value exists.

  Type "double"; specifies the value.

- The following five fields were changed from type long to the *OptionalDoubleEntity* class:
  - MinimalServiceComponentLevelTarget
  - MaximalServiceComponentLevelTarget
  - YellowThreshold
  - OrangeThreshold
  - RedThreshold

Remarks

Prior to version 7.0 Service Pack 1, the above five fields could not be given a null value or a non-integer value. If for any reason they were given a null or a non-integer value, the API automatically assigned them the value zero instead.

As of version 7.0 Service Pack 1, these five fields were modified and can now be assigned null or non-integer values.

Example (C#)

When Reading

Instead of

```
double target = MinimalServiceComponentLevelTarget;
```

Use

```
double target;
    If (MinimalServiceComponentLevelTarget != null)
            {
            If (MinimalServiceComponentLevelTarget.IsValueLess == true)
            target = null;
            Else
            target = MinimalServiceComponentLevelTarget.Value;
            }
            Else
    target = null;
```

When Writing

Previously

```
MinimalServiceComponentLevelTarget = 99.999
```

Currently

```
MinimalServiceComponentLevelTarget = new OptionalDoubleEntity();
MinimalServiceComponentLevelTarget.IsValueLess = true;
MinimalServiceComponentLevelTarget.Value = 99.999;
```

Changes to the API Interface

The following change was made to the API Interface.

- A new field called *ParameterOrder* was added to the *ParameterData* structure. This affects the following methods:
  - *CreateContractMetric*
  - *CreateContractMetricParameter*
  - *AddContractParameter*
  - *CreateContractTemplateMetric*
  - *CreateContractTemplateMetricParameter*
  - *CreateContractTemplateParameter*
  - *CreateServiceDefinitionMetric*
  - *CreateServiceDefinitionMetricParameter*

- *CreateServiceDefinitionParameter*
- *GetContractMetricDetails*
- *GetContractParameters*
- *GetContractTemplateMetricDetails*
- *GetContractTemplateParameters*
- *GetServiceDefinitionParameters*
- *GetServiceDefinitionMetricDetails*
- *UpdateContractMetricParameter*
- *UpdateContractParameter*
- *UpdateContractMetricDetails*
- *UpdateContractTemplateParameter*
- *UpdateContractTemplateMetricParameter*
- *UpdateContractTemplateMetricDetails*
- *UpdateServiceDefinitionMetricParameter*
- *UpdateServiceDefinitionMetricDetails*
- *UpdateServiceDefinitionParameter*

Remarks

Various API methods return a set of parameters containing a structure without any specific internal order.

The *CreateContractMetric* and *UpdateContractMetricDetails* methods (as well as others) get the *MetricData* structure with all metric parameters. As of version 7.0 Service Pack 1 the user can set the parameter order in this structure.

Note: The API does not check for consistency (more than one member can have the same *ParameterOrder* value. In this case, there is no way to know the precise internal order of members with a duplicated order).

Example

The *GetContractParameters* method returns a set of parameter data in which the new *ParameterOrder* field is assigned a value (e.g., 0,1,2, ..) If we want to change the order of any of the data we must change the value of the *ParameterOrder* field according to the desired order.

API Infrastructure

The API infrastructure for CA Business Service Insight consists of:

- Security (see page 185)
- Data Validation (see page 187)
- Error Handling (see page 187)

The following sections describe these three items.

Security

Overview of Security

CA Business Service Insight uses WCF (Windows Communication Foundation) with message security. (More information about WCF can be found at http://msdn.microsoft.com/en-us/netframework/aa663324.aspx.)

The username, organization, and password are sent to authorize every requested operation. These three items are encrypted using an X.509 certificate. (More information about X.509 certificates can be found at http://en.wikipedia.org/wiki/X.509.)

How to build the user name is described in Validating Installation of the Certificate (see page 186).

Creating and Installing the Certificate

The system administrator creates an X.509 certificate. This certificate should be registered with the CA Business Service Insight server, preferably in the LocalMachine/TrustedPeople store.

Afterward, CA Business Service Insight needs to know where this certificate resides. Pointing Cloud Insight to the registered certificate is done from the webroot/API/Web.config XML file. In this file, in the configuration/system.serviceModel/behaviors/serviceBehaviors/behavior/serviceCredentials/serviceCertificate node, change the values in the following four fields to point to the registered certificate:

- findValue
- storeLocation
- storeName
- x509FindType

CA Business Service Insight can then publish the public portion of the certificate in the WSDL (Web Server Definition Language) data, to enable encryption of data transmission.

Validating Installation of the Certificate

To validate the installation of the certificate:

1. Server side:
   - Install the certificate on a CA Business Service Insight server, as described in Creating and Installing the Certificate (see page 186).
   - If applicable, navigate to http//server/api/contract.svc. The *OK* screen for the Contract server should appear.
   - If applicable, navigate to http//server/api/reporting.svc. The *OK* screen for the Reporting server should appear.
   - If applicable, navigate to http//server/api/portfolio.svc. The *OK* screen for the Portfolio server should appear.
2. Client side:

Note: There are various tools for creating a proxy, such as Microsoft Visual Studio or a third party. The client will select the tool to use.

For the proxy username, use the format:
<your CA Business Service Insight username>@<your CA Business Service Insight organization>

For example: johndoe@YourCompany

For the password, use your CA Business Service Insight password.

- If applicable, create a service client to http//server/api/contract.svc.
- If applicable, create a service client to http//server/api/portfolio.svc.

Data Validation

Data errors can be caused by invalid authentication or authorization, bad data (during data validation), or business errors.

During data validation, the data is checked for possible errors such as incorrect data type, impermissible values, and invalid string lengths.

A data error of any kind gives the user a SOAP Fault. The details of the error can be found in the Reason field of the SOAP Fault. (More information about SOAP Fault and the Reason field can be found at http://msdn.microsoft.com/en-us/library/ms189538.aspx.)

Error Handling

The error handling for CA Business Service Insight is proprietary. In case of an error, one of CA Business Service Insight's fault types will be returned.

The error types, and the fault types as which they are exposed, are:

- Data validation errors
- Authorization and authentication errors
- Communication errors (timeout)
- Business errors
- All other errors ("safety net" for unhandled errors)

Error Handling Demo Service

The CA application comes with an Error Handling Demo service. This service can be found at webroot/API/ErrorHandlingDemo.svc.

The Error Handling Demo service has six error-producing methods, all of which return no value (void):

- CreateNoError() – To test “no error” situations.
- YieldValidatorError(int number) – To test data validation errors.
- DemandViewAlertPermissions() – To test authorization and authentication errors.
- Wait(int seconds) – To test communication errors (timeout).
- YieldEntityExistsFault() – To test business errors.
- ThrowException() – To test all other errors.

These methods create errors that represent all the error types in CA Business Service Insight. These methods are useful when creating and testing a proxy that can expect and handle bounced faults.

Open API Interface Methods

The Open API interface methods can be divided into four parts:

- Contract Interface (see page 191)
- Portfolio Interface (see page 223)
- Repository Interface (see page 275)
- Configuration Interface (see page 188)

Configuration Interface

The Configuration Interface methods are listed in the following table.

GetResources (see page 189)	GetResourceTypes (see page 190)
GetResourceGroups (see page 189)	GetEventTypes (see page 191)

Resource Methods

GetResources

Description

This method gets all resources in the system that match the search string.

Syntax

EntityListData GetResources (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for.

Return Value

The resources, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetResourceGroups

Description

This method gets all resource groups in the system that match the search string.

Syntax

EntityListData GetResourceGroups (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for.

Return Value

The resource groups, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetResourceTypes

Description

This method gets all resource types in the system that match the search string.

Syntax

EntityListData GetResourceTypes (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for.

Return Value

The resource types, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Event Methods

GetEventTypes

Description

This method gets all event types in the system that match the search string.

Syntax

EntityListData GetEventTypes (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for.

Return Value

The resource types, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Contract Interface

The Contract Interface methods are listed in the following eight tables.

Contract Creation Methods	
CreateContract (see page 193)	CreateContractFromServiceDefinition (see page 194)
CreateContractFromContractTemplate (see page 193)	DuplicateContract (see page 195)

Contract Details Methods	
GetContractDetails (see page 196)	GetContractTimeSlots (see page 198)
GetContracts (see page 196)	GetContractCustomAttributes (see page 198)
UpdateContractDetails (see page 197)	

Contract Metric Registration Methods

AddContractMetricRegistration (see page 199)	DeleteContractMetricRegistration (see page 200)
UpdateContractMetricRegistration (see page 200)	GetContractMetricRegistrations (see page 201)

Contract Metric Notes Methods

GetContractMetricNotes (see page 206)	UpdateContractMetricNotes (see page 206)

Contract Metrics Methods

GetContractMetrics (see page 202)	UpdateContractMetricDetails (see page 204)
DeleteContractMetric (see page 202)	GetContractMetricDetails (see page 204)
CreateContractMetric (see page 203)	AddMetricsToContract (see page 205)

Contract Parameter Methods

GetContractParameters (see page 207)	DeleteContractMetricParameter (see page 210)
DeleteContractParameter (see page 208)	UpdateContractMetricParameter (see page 211)
AddContractParameter (see page 209)	CreateContractMetricParameter (see page 212)
UpdateContractParameter (see page 210)	

Contract Related Entities Methods

GetContractTypes (see page 212)	GetMetricMainIndicatorTypes (see page 214)
GetContractParties (see page 213)	GetMetricSectionsByType (see page 215)
GetMetricTypes (see page 214)	

Contract Service Relationship Methods

GetContractByServiceComponent (see page 216)	AddServiceComponentToContract (see page 216)

Contract Versioning and Deleting Methods

CommitContract (see page 217)	DeleteContractVersion (see page 219)
CreateNewContractVersion (see page 218)	

Contract Workflow Methods	
ContractWorkflowRequestApproval (see page 220)	ContractWorkflowApprove (see page 222)
ContractWorkflowCancelApprovalRequest (see page 221)	ContractWorkflowDeny (see page 222)

Contract Creation Methods

CreateContract

Description

This method creates a contract from the input data.

Syntax

long CreateContract (ContractData contractData);

Parameters

Name	Type	Description
contractData	ContractData	The data to use to create the contract.

Return Value

The ID of the new contract, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ContractData object provided for this method should be filled with an ID for every sub-entity of which it consists (for example, the contract's corresponding primary contract party).

CreateContractFromContractTemplate

Description

This method creates a contract from the input data and a contract template.

Syntax

long CreateContractFromContractTemplate (ContractData cData, long contractTemplateFolderItemID);

Parameters

Name	Type	Description
cData	ContractData	The contract data to use.
contractTemplateFolderItemID	long	The ID of the contract template to use.

Return Value

The ID of the new contract, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ContractData object provided for this method should be filled with an ID for every sub-entity of which it consists (for example, the contract's corresponding primary contract party).

CreateContractFromServiceDefinition

Description

This method creates a contract from a collection of metrics associated with a specific service definition.

Syntax

long CreateContractFromServiceDefinition (ContractData contractData, long[] MetricIDs);

Parameters

Name	Type	Description
contractData	ContractData	The data to use for creating the contract.
MetricIDs	long[]	An array of IDs of the selected service definition metrics.

Return Value

The ID of the new contract, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ContractData object provided for this method should be filled with an ID for every sub-entity of which it consists (for example, the contract's corresponding primary contract party).

DuplicateContract

Description

This method duplicates a contract.

Syntax

long DuplicateContract (ContractData cData, long ContractVersionID);

Parameters

Name	Type	Description
cData	ContractData	The contract data to use.
ContractVersionID	long	The ID of the contract version to duplicate.

Return Value

The ID of the new contract, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ContractData object provided for this method should be filled with an ID for every sub-entity that it contains (for example, the contract's corresponding primary contract party).

Contract Details Methods

GetContractDetails

Description

This method gets the details for the specified contract version.

Syntax

ContractData GetContractDetails (long ContractVersionID);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.

Return Value

The contract details, as type ContractData.

Remarks

The data returned is already filled with both the numerical ID and the textual name for every sub-entity of which it consists.

GetContracts

Description

This method gets a list of all contracts in the system for which the name matches the search string.

Syntax

EntityListData GetContracts (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for.

Return Value

The contracts, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

UpdateContractDetails

Description

This method updates all of the details of the specified contract.

Syntax

void UpdateContractDetails (ContractData contractData, long ContractVersionID);

Parameters

Name	Type	Description
contractData	ContractData	The new data for the contract.
ContractVersionID	long	The ID of the contract version to update.

Return Value

None.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ContractData object replaces the data of the existing contract, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding primary contract party).

GetContractTimeSlots

Description

This method gets a list of all time slots of the specified contract version for which the name matches the search string.

Syntax

EntityListData GetContractTimeSlots (long ContractVersionId, string searchText);

Parameters

Name	Type	Description
ContractVersionId	long	The ID of the contract version.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The time slots, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetContractCustomAttributes

Description

This method gets all custom attributes for the specified contract version.

Syntax

CustomAttributesCollection GetContractCustomAttributes (long SlaVersionID);

Parameters

Name	Type	Description
SlaVersionID	long	The ID of the contract version.

Return Value

The custom attributes, as type CustomAttributesCollection.

Remarks

None.

Contract Metric Registration Methods

AddContractMetricRegistration

Description

This method creates an event registration for the specified contract metric.

Syntax

void AddContractMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to create the registration.

Return Value

None.

Remarks

The ID of the newly created registration can be obtained as part of the metric's registration collection, by invoking GetContractMetricRegistrations (see page 201).

UpdateContractMetricRegistration

Description

This method updates the specified event registration of the specified contract metric.

Syntax

void UpdateContractMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to update the registration.

Return Value

None.

Remarks

The ID of the registration to update is contained in the MetricRegistrationData type class (see MetricRegistrationData (see page 296)).

DeleteContractMetricRegistration

Description

This method deletes the specified event registration of the specified contract metric.

Syntax

void DeleteContractMetricRegistration (long metricID, long registrationID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
registrationID	long	The ID of the registration.

Return Value

None.

Remarks

None.

GetContractMetricRegistrations

Description

This method gets all event registrations of the specified contract metric.

Syntax

MetricRegistrationsCollection GetContractMetricRegistrations (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The event registrations of the metric, as type MetricRegistrationsCollection.

Remarks

None.

Contract Metrics Methods

GetContractMetrics

Description

This method gets a list of all contract metrics of the specified contract version for which the name matches the search string.

Syntax

EntityListData GetContractMetrics (long ContractVersionID, string searchText);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The contract metrics, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

DeleteContractMetric

Description

This method deletes a contract metric.

Syntax

void DeleteContractMetric (long MetricId);

Parameters

Name	Type	Description
MetricID	long	The ID of the metric to delete.

Return Value

None.

Remarks

None.

CreateContractMetric

Description

This method creates a contract metric for the specified contract version.

Syntax

long CreateContractMetric (long contractVersionID, MetricData metricData);

Parameters

Name	Type	Description
ContractVersionId	long	The ID of the contract version.
metricData	MetricData	The data to use to create the new metric.

Return Value

The ID of the new metric, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

Refer to MetricData (see page 295).

UpdateContractMetricDetails

Description

This method updates a specific metric's details of the specified contract version.

Syntax

void UpdateContractMetricDetails (long contractVersionID, MetricData metricData);

Parameters

Name	Type	Description
ContractVersionId	long	The ID of the contract version.
metricData	MetricData	The data to use to update the metric.

Return Value

None.

Remarks

The ID of the metric to update is contained in the MetricData type class (see MetricData Classes (see page 295)).

The method validates the data sent to it and may throw a validation exception upon failure.

GetContractMetricDetails

Description

This method gets all details for the specified contract metric.

Syntax

MetricData GetContractMetricDetails (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The metric details, as type MetricData.

Remarks

The data returned is already filled with both the numerical ID and the text name for every sub-entity that it contains.

AddMetricsToContract

Description

This method copies existing metrics in the system to the specified contract version.

Syntax

void AddMetricsToContract (long contractVersionItemID, long[] MetricIDs, bool forceAdd);

Parameters

Name	Type	Description
contractVersionItemId	long	The ID of the contract version.
MetricIDs	long[]	An array of IDs of the selected metrics to add.
forceAdd	bool	Specifies whether to force the addition.

Return Value

The copied metrics may belong to another contract, service definition, or contract template.

If the addition fails due to existing metrics with similar names, or mismatched effective dates, it is possible to set the **forceAdd** argument to **true**, in order to perform the addition while automatically renaming the metrics.

Remarks

None.

Get Contract Metric Notes Methods

GetContractMetricNotes

Description

This method gets the details of the metric notes in a contract.

Syntax

MetricNotesData GetContractMetricNotes(long metricID);

Parameters

Name	Type	Description
metrictID	long	The metric's ID.

Return Value

The metric notes details, as type MetricNotesData.

Remarks

None.

UpdateContractMetricNotes

Description

This method updates the metric notes in a contract.

Syntax

void UpdateContractMetricNotes(long metricID, MetricNotesData data);

Parameters

Name	Type	Description
metricID	long	The Metric's ID.
data	MetricNotesData	The data used to update notes of this metric.

Return Value

None.

Remarks

None.

Contract Parameter Methods

GetContractParameters

Description

This method gets all contract level parameters for the specified contract version.

Syntax

ParametersCollection GetContractParameters (long ContractVersionID);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.

Return Value

The parameters of the contract version, as type ParametersCollection.

Remarks

None.

GetContractStatus

Description

This method gets a list of all contracts in the system for which the name matches the search string.

Syntax

string GetContractStatus (long ContractVersionID)

Parameters

Name	Type	Description
ContractVersionID	long	The contract version ID.

Return Value

Returns the status in English UPPERCASE letters (as it exists in the data base).

Remarks

None.

DeleteContractParameter

Description

This method deletes the specified contract level parameter of the specified contract version.

Syntax

void DeleteContractParameter (long ContractVersionID, long ParameterID);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
ParameterID	long	The ID of the parameter to delete.

Return Value

None.

Remarks

None.

CreateContractParameter

Description

This method creates a contract level parameter for the specified contract version.

Syntax

void AddContractParameter (long ContractVersionID, ParameterData parameterData);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained by invoking *GetContractParameters* (see page 207).

UpdateContractParameter

Description

This method updates a specific contract level parameter of the specified contract version.

Syntax

void UpdateContractParameter (long ContractVersionID, ParameterData parameterData);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

DeleteContractMetricParameter

Description

This method deletes the specified parameter of the specified contract metric.

Syntax

void DeleteContractMetricParameter (long MetricId, long ParameterId);

Parameters

Name	Type	Description
MetricID	long	The ID of the metric to delete.
ParameterID	long	The ID of the parameter to delete.

Return Value

None.

Remarks

None.

UpdateContractMetricParameter

Description

This method updates a specific parameter of the specified contract metric.

Syntax

void UpdateContractMetricParameter (long MetricId, ParameterData parameterData);

Parameters

Name	Type	Description
MetricID	long	The ID of the metric.
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

CreateContractMetricParameter

Description

This method creates a parameter for the specified contract metric.

Syntax

void UpdateContractMetricParameter (long MetricId, ParameterData parameterData);

Parameters

Name	Type	Description
MetricID	long	The ID of the metric.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained as part of the metric's parameters collection, in the metric's details, by invoking *GetContractMetricDetails* (see page 204).

Contract Related Entities Methods

GetContractTypes

Description

This method gets a list of all contract types in the system for which the name matches the search string.

Syntax

EntityListData GetContractTypes (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The contract types, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetContractParties

Description

This method gets a list of all contract parties in the system for which the name matches the search string.

Syntax

EntityListData GetContractParties (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The contract parties, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetMetricTypes

Description

This method gets a list of all metric types in the system for which the name matches the search string.

Syntax

EntityListData GetMetricTypes (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The metric types, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetMetricMainIndicatorTypes

Description

This method gets a list of all metric main indicator types in the system for which the name matches the search string.

Syntax

EntityListData GetMetricMainIndicatorTypes (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The IDs and names of the system's metric main indicator types, as type EntityListData.

Remarks

The metric's main indicator type indicates the entity that the metric is associated with.

Currently there are two options: either the metric is associated with a specific service component, or it is associated with its parent contract entity.

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetMetricSectionsByType

Description

This method gets a list of all metric sections in the system that match the specified metric type.

Syntax

EntityListData GetMetricSectionsByType (long MetricTypeId);

Parameters

Name	Type	Description
MetricTypeId	long	The ID of the metric type.

Return Value

The matching metric sections, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

CCE--INS--Contract Service Relationship Methods

AddServiceToContract

Description

This method adds the service to the contract as an included service.

Syntax

void AddServiceComponentToContract (long contractVersionID, long serviceComponentID);

Parameters

Name	Type	Description
contractVersionID	long	The contract version ID.
serviceComponentID	long	The service component ID.

Return Value

None.

Remarks

None.

GetContractByServiceComponent

Description

This method gets a list of all contracts belonging to the service component.

Syntax

A typical syntax for this method is shown next.

EntityListData GetContractsByServiceComponent (long serviceComponentID);

Parameters

Name	Type	Description
serviceComponentID	long	The service component ID.

Return Value

The contracts, as type EntityListData.

Remarks

The ServiceComponentID specifies the included services.

An EntityListData object is returned as a collection of EntityListItemData items and includes the name and ID of every object.

Contract Versioning and Deleting Methods

CommitContract

Description

This method commits the specified contract version changes, making it effective for data calculation.

Syntax

ActionInfoCollection CommitContract (long ContractVersionID, string Justification, bool forceCommit);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
Justification	string	Text describing the reasons for committing the contract changes, and any additional comments.
forceCommit	bool	Specifies whether to force the commit.

Return Value

An ActionInfoCollection object, which is a collection of zero or more ActionInfo items with information about the failure or success of the commit process.

Each ActionInfo item consists of:

- ActionInfoSeverity - An enumeration of **Error**, **Warning**, **Info**, or **IneffectiveModulesMessages**.
- Info - An information string.
- InfoCode - The system's identifier for the specific message.

Remarks

If the value of **forceCommit** is set to **false**, the contract is not committed at all; a notification will be issued if there is a Warning. If the **forceCommit** argument is set to to **true**, the contract will be committed despite any warnings.

CreateNewContractVersion

Description

This method creates a new contract version.

Syntax

long CreateNewContractVersion (ContractData cData, long ContractVersionID);

Parameters

Name	Type	Description
cData	ContractData	The data to use to create the new contract version.
ContractVersionID	long	The ID of the current contract version.

Return Value

The ID of the new contract version, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ContractData object replaces the data of the existing contract version, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding primary contract party).

DeleteContractVersion

Description

This method deletes the specified contract version.

Syntax

ActionInfoCollection DeleteContractVersion (long contractVersionID, bool forceDelete);

Parameters

Name	Type	Description
contractVersionID	long	The ID of the contract version to delete.
forceDelete	bool	Specifies whether to force the deletion.

Return Value

An ActionInfoCollection object, which is a collection of zero or more ActionInfo items with information about the failure or success of the deletion process.

Each ActionInfo item consists of:

- ActionInfoSeverity - An enumeration of **Error**, **Warning**, **Info**, or **IneffectiveModulesMessages**.
- Info - An information string.
- InfoCode - The system's identifier for the specific message.

Remarks

In case the deletion action fails and the system returns only warning messages, but not errors, it is possible to set the **forceDelete** argument to **true**, in order to perform the commit despite the warnings.

Contract Workflow Methods

ContractWorkflowRequestApproval

Description

This method sends an e-mail for an approval request as part of a contract version's workflow.

The email is automatically sent to the address that is set in the address field of the contract's approver.

Syntax

void ContractWorkflowRequestApproval (long ContractVersionID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
WorkFlowRemarks	string	Any remarks regarding the version and its approval request.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts preferences should be provided with the address of an accessible, valid mail server.

ContractWorkflowCancelApprovalRequest

Description

This method sends an e-mail cancelling a request for approval of a contract version's workflow.

The e-mail is automatically sent to the address that is set in the address field of the contract's approver.

Syntax

void ContractWorkflowCancelApprovalRequest (long ContractVersionID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
WorkFlowRemarks	string	Any remarks regarding the cancellation.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts configuration should be provided with the address of an accessible, valid mail server.

ContractWorkflowApprove

Description

This method approves a contract version's workflow.

Syntax

void ContractWorkflowApprove (long ContractVersionID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
WorkFlowRemarks	string	Any remarks regarding the approval.

Return Value

None.

Remarks

None.

ContractWorkflowDeny

Description

This method denies the approval of a contract version's workflow.

Syntax

void ContractWorkflowDeny (long ContractVersionID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractVersionID	long	The ID of the contract version.
WorkFlowRemarks	string	Any remarks regarding the denial.

Return Value

None.

Remarks

None.

Portfolio Interface

The Portfolio Interface methods are listed in the following three tables.

Portfolio Methods	
GetPortfolios (see page 225)	GetCatalogItems (see page 226)
GetPortfolioItems (see page 226)	

Contract Template Methods	
GetContractTemplateFolderItems (see page 230)	CreateContractTemplateMetricParameter (see page 240)
GetContractTemplateParameters (see page 244)	AddMetricsToContractTemplate (see page 240)
DeleteContractTemplateParameter (see page 244)	AddContractTemplateMetricRegistration (see page 241)
CreateContractTemplateParameter (see page 245)	UpdateContractTemplateMetricRegistration (see page 242)
UpdateContractTemplateParameter (see page 246)	DeleteContractTemplateMetricRegistration (see page 242)
GetContractTemplateDetails (see page 228)	GetContractTemplateMetricRegistrations (see page 243)
UpdateContractTemplateDetails (see page 228)	ContractTemplateWorkflowRequestApproval (see page 247)
GetContractTemplateTimeSlots (see page 229)	ContractTemplateWorkflowCancelApprovalRequest (see page 248)
GetContractTemplateCustomAttributes (see page 230)	ContractTemplateWorkflowApprove (see page 248)
GetContractTemplateMetrics (see page 234)	ContractTemplateWorkflowDeny (see page 249)
DeleteContractTemplateMetric (see page 236)	DeleteContractTemplate (see page 231)

CreateContractTemplateMetric (see page 236)	DeleteContractTemplateLatestVersion (see page 232)
GetContractTemplateMetricNotes (see page 232)	UpdateContractTemplateMetricNotes (see page 233)
GetContractTemplateMetricDetails (see page 237)	CreateNewContractTemplateVersion (see page 250)
UpdateContractTemplateMetricDetails (see page 238)	CommitContractTemplate (see page 251)
DeleteContractTemplateMetricParameter (see page 238)	CreateContractTemplate (see page 227)
UpdateContractTemplateMetricParameter (see page 239)	

ServiceDefinition Methods

GetServiceDefinitionParameters (see page 251)	CreateServiceDefinitionMetricParameter (see page 264)
DeleteServiceDefinitionParameter (see page 252)	AddMetricsToServiceDefinition (see page 264)
CreateServiceDefinitionParameter (see page 253)	AddServiceDefinitionMetricRegistration (see page 266)
UpdateServiceDefinitionParameter (see page 253)	UpdateServiceDefinitionMetricRegistration (see page 267)
GetServiceDefinitionDetails (see page 256)	DeleteServiceDefinitionMetricRegistration (see page 268)
UpdateServiceDefinitionDetails (see page 257)	GetServiceDefinitionMetricRegistrations (see page 268)
GetServiceDefinitionTimeSlots (see page 258)	ServiceDefinitionWorkflowRequestApproval (see page 269)
GetServiceDefinitionCustomAttributes (see page 258)	ServiceDefinitionWorkflowCancelApprovalRequest (see page 270)
GetServiceDefinitionMetrics (see page 259)	ServiceDefinitionWorkflowApprove (see page 270)
DeleteServiceDefinitionMetric (see page 260)	ServiceDefinitionWorkflowDeny (see page 271)
CreateServiceDefinitionMetric (see page 261)	DeleteServiceDefinition (see page 272)
GetServiceDefinitionFolderItems (see page 254)	GetServiceDefinitionFolderItemsByServiceID (see page 255)

GetServiceDefinitionMetricDetails (see page 262)	DeleteServiceDefinitionLatestVersion (see page 272)
UpdateServiceDefinitionMetricDetails (see page 262)	CreateNewServiceDefinitionVersion (see page 273)
GetServiceDefinitionMetricNotes (see page 265)	UpdateServiceDefinitionMetricNotes (see page 266)
DeleteServiceDefinitionMetricParameter (see page 263)	CommitServiceDefinition (see page 273)
UpdateServiceDefinitionMetricParameter (see page 263)	CreateServiceDefinition (see page 274)

Portfolio Methods

GetPortfolios

Description

This method gets a list of all portfolios in the system for which the name matches the search string.

Syntax

EntityListData GetPortfolios (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The portfolios, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetPortfolioItems

Description

This method gets all items for the specified portfolio.

Syntax

PortfolioEntityCollection GetPortfolioItems (long PortfolioId);

Parameters

Name	Type	Description
PortfolioId	long	The portfolio ID.

Return Value

The items of the portfolio, as type PortfolioEntityCollection.

Remarks

None.

GetCatalogItems

Description

This method gets all items for the specified catalog.

Syntax

PortfolioEntityCollection GetCatalogItems (long CatalogId);

Parameters

Name	Type	Description
CatalogId	long	The catalog ID.

Return Value

The items of the catalog, as type PortfolioEntityCollection.

Remarks

None.

ContractTemplate Methods

ContractTemplate Creation Methods

CreateContractTemplate

Description

This method creates a contract template from the input data.

Syntax

long CreateContractTemplate (ContractTemplateData contractTemplateData, PortfolioContainerType ContractTemplateContainerType, long ContractTemplateContainerID);

Parameters

Name	Type	Description
contractTemplateData	ContractTemplateData	The data to use to create the new contract template.
ContractTemplateContainerType	PortfolioContainerType	The container type of the new contract template.
ContractTemplateContainerID	long	The container ID of the new contract template.

Return Value

The ID of the new contract template, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ContractTemplateData object provided for this method should be filled with an ID for every sub-entity of which it consists (for example, the contract's corresponding primary contract party).

ContractTemplate Details Methods

GetContractTemplateDetails

Description

This method gets all details for the specified contract template.

Syntax

ContractTemplateData GetContractTemplateDetails (long contractTemplateFolderItemId);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

The contract template details, as type ContractTemplateData.

Remarks

The data returned is already filled with both the numerical IDs and the textual name of every sub-entity of which it consists.

UpdateContractTemplateDetails

Description

This method updates the details for the specified contract template.

Syntax

void UpdateContractTemplateDetails (ContractTemplateData contractTemplateData, long contractTemplateFolderItemId);

Parameters

Name	Type	Description
contractTemplateData	ContractTemplateData	The data to use to update the contract template.
contractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

None.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ContractTemplateData object replaces the data of the existing contract template, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding service components).

GetContractTemplateTimeSlots

Description

This method gets a list of all time slots of the specified contract template for which the name matches the search string.

Syntax

EntityListData GetContractTemplateTimeSlots (long contractTemplateFolderItemId, string searchText;

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The time slots, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetContractTemplateCustomAttributes

Description

This method gets all custom attributes for the specified contract template.

Syntax

CustomAttributesCollection GetContractTemplateCustomAttributes (long contractTemplateFolderItemId);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

The custom attributes, as type CustomAttributesCollection.

Remarks

None.

GetContractTemplateFolderItems

Description

This method gets a list of contract template folder items in the system for which the name matches the search string.

Syntax

EntityListData GetContractTemplateFolderItems(string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and can be left empty. If used, it can contain an asterisk as a wildcard for the search.

Return Value

The list of the contract template folder items, as type EntityListData which contains the name and the folder item ID of contract templates.

Remarks

None.

ContractTemplate Detach/Delete Methods

DeleteContractTemplate

Description

This method deletes the specified contract template.

Syntax

void DeleteContractTemplate (long contractTemplateFolderItemId);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

None.

Remarks

None.

DeleteContractTemplateLatestVersion

Description

This method deletes the latest version of the specified contract template.

Syntax

void DeleteContractTemplateLatestVersion (long contractTemplateFolderItemId);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

None.

Remarks

None.

Contract Template Metric Notes Methods

GetContractTemplateMeetricNotes

Description

This method get the details of the metric notes in a contract.

Syntax

MetricNotesData GetContractTemplateMetricNotes(long metricID);

Parameters

Name	Type	Description
metricID	long	The Metric's ID.

Return Value

The metric notes details, as type MetricNotesData.

Remarks

None.

UpdateContractTemplateMetricNotes

Description

This method updates the metric notes in a contract template.

Syntax

void UpdateContractTemplateMetricNotes(long metricID, MetricNotesData data);

Parameters

Name	Type	Description
metricID	long	The Metric's ID.
data	MetricNotesData	The data used to update notes of this metric.

Return Value

None.

Remarks

None.

ContractTemplate Metrics Methods

GetContractTemplateMetrics

Description

This method gets a list of all metrics of the specified contract template for which the name matches the search string.

Syntax

EntityListData GetContractTemplateMetrics (long contractTemplateFolderItemId, string searchText);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The contract template metrics, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

CCE—INS—GetContractTemplateLastCommittedMetrics

Description

This method gets a list of metrics of the last committed version of the specified contract template for which the name matches the search string.

Syntax

EntityListData GetContractTemplateLastCommittedMetrics (long contraFolderItemID, string searchText);

Parameters

Name	Type	Description
contraFolderItemID	long	The ID of the contract template item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The metrics of the last, committed Contract Template, as type EntityListData.

Note: Returns metrics only if there is a committed version.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Note: The status (effective, not effective, pending) does not have an affect; the sole consideration is whether the template is "committed" or not committed.

Note: Throws an error when the Contract Template has no committed version.

DeleteContractTemplateMetric

Description

This method deletes the specified contract template metric.

Syntax

void DeleteContractTemplateMetric (long MetricId);

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.

Return Value

None.

Remarks

None.

CreateContractTemplateMetric

Description

This method creates a metric for the specified contract template.

Syntax

long CreateContractTemplateMetric (long contractTemplateFolderItemId, MetricData metricData);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
metricData	MetricData	The data to use to create the new metric.

Return Value

The ID of the new metric, as type long.

Remarks

None.

GetContractTemplateMetricDetails

Description

This method gets all details for the specified contract template metric.

Syntax

MetricData GetContractTemplateMetricDetails (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The metric details, as type MetricData.

Remarks

The data returned is already filled with both the numerical ID and the text name for every sub-entity that it contains.

UpdateContractTemplateMetricDetails

Description

This method updates metric details for the specified contract template.

Syntax

void UpdateContractTemplateMetricDetails (long contractTemplateFolderItemId, MetricData metricData);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
metricData	MetricData	The data to use to update the metric details.

Return Value

None.

Remarks

None.

DeleteContractTemplateMetricParameter

Description

This method deletes the specified parameter of the specified contract template metric.

Syntax

void DeleteContractTemplateMetricParameter (long MetricId, long ParameterId);

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.
ParameterId	long	The ID of the parameter.

Return Value

None.

Remarks

None.

UpdateContractTemplateMetricParameter

Description

This method updates a specific parameter of the specified contract template metric.

Syntax

void UpdateContractTemplateMetricParameter (long MetricId, ParameterData parameterData;

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

CreateContractTemplateMetricParameter

Description

This method creates a parameter for the specified metric.

Syntax

void CreateContractTemplateMetricParameter (long MetricId, ParameterData parameterData;

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained as part of the metric's parameters collection, in the metric's details, by invoking *GetContractTemplateMetricDetails* (see page 237).

AddMetricsToContractTemplate

Description

This method copies existing metrics in the system to the specified contract template.

Syntax

void AddMetricsToContractTemplate (long contractTemplateFolderItemID, long[] MetricIDs, bool forceAdd);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
MetricIDs	long[]	An array of IDs of the selected metrics to add.
forceAdd	bool	Specifies whether to force the addition.

Return Value

The copied metrics may belong to another contract, service definition, or contract template.

If the addition fails due to existing metrics with similar names, or mismatched effective dates, it is possible to set the **forceAdd** argument to **true**, in order to perform the addition while automatically renaming the metrics.

Remarks

None.

ContractTemplate Metric Registration Methods

AddContractTemplateMetricRegistration

Description

This method creates an event registration for the specified contract template metric.

Syntax

void AddContractTemplateMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to create the registration.

Return Value

None.

Remarks

The ID of the newly created registration can be obtained as part of the metric's registration collection, by invoking GetContractTemplateMetricRegistrations (see page 243).

UpdateContractTemplateMetricRegistration

Description

This method updates the specified event registration of the specified contract template metric.

Syntax

void UpdateContractTemplateMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to update the registration.

Return Value

None.

Remarks

The ID of the registration to update is contained in the MetricRegistrationData type class (see MetricRegistrationData (see page 296)).

DeleteContractTemplateMetricRegistration

Description

This method deletes the specified event registration of the specified contract template metric.

Syntax

void DeleteContractTemplateMetricRegistration (long metricID, long registrationID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
registrationID	long	The ID of the registration.

Return Value

None.

Remarks

None.

GetContractTemplateMetricRegistrations

Description

This method gets all event registrations of the specified contract template metric.

Syntax

MetricRegistrationsCollection GetContractTemplateMetricRegistrations (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The event registrations of the metric, as type MetricRegistrationsCollection.

Remarks

None.

ContractTemplate Parameters Methods

GetContractTemplateParameters

Description

This method gets all contract level parameters for the specified contract template.

Syntax

ParametersCollection GetContractTemplateParameters (long contractTemplateFolderItemId);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

The contract template parameters, as type ParametersCollection.

Remarks

None.

DeleteContractTemplateParameter

Description

This method deletes the specified contract level parameter of the specified contract template.

Syntax

void DeleteContractTemplateParameter (long contractTemplateFolderItemId, long ParameterID);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
ParameterID	long	The ID of the parameter to delete.

Return Value

None.

Remarks

None.

CreateContractTemplateParameter

Description

This method creates a contract level parameter for the specified contract template.

Syntax

void CreateContractTemplateParameter (long contractTemplateFolderItemId, ParameterData parameterData);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained by invoking *GetContractTemplateParameters* (see page 244).

UpdateContractTemplateParameter

Description

This method updates a specific contract level parameter for the specified contract template.

Syntax

void UpdateContractTemplateParameter (long contractTemplateFolderItemId, ParameterData parameterData);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract template folder item.
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

ContractTemplate Workflow Methods

ContractTemplateWorkflowRequestApproval

Description

This method sends an e-mail for an approval request as part of a contract template version's workflow.

The e-mail is automatically sent to the address that is set in the address field of the contract template's approver.

Syntax

void ContractTemplateWorkflowRequestApproval (long ContractTemplateFolderItemId, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.
WorkFlowRemarks	string	Any remarks regarding the version and its approval request.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts preferences should be provided with the address of an accessible, valid mail server.

ContractTemplateWorkflowCancelApprovalRequest

Description

This method send an e-mail cancelling a request for approval of a contract template versions's workflow.

Syntax

void ContractTemplateWorkflowCancelApprovalRequest (long ContractTemplateFolderItemId, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.
WorkFlowRemarks	string	Any remarks regarding the cancellation.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts configuration should be provided with the address of an accessible, valid mail server.

ContractTemplateWorkflowApprove

Description

This method approves a contract template version's workflow.

Syntax

void ContractTemplateWorkflowApprove (long ContractTemplateFolderItemId, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.
WorkFlowRemarks	string	Any remarks regarding the approval.

Return Value

None.

Remarks

None.

ContractTemplateWorkflowDeny

Description

This method denies a contract template version's workflow.

Syntax

void ContractTemplateWorkflowDeny (long ContractTemplateFolderItemId, string WorkFlowRemarks);

Parameters

Name	Type	Description
ContractTemplateFolderItemId	long	The ID of the contract template folder item.
WorkFlowRemarks	string	Any remarks regarding the denial.

Return Value

None.

Remarks

None.

ContractTemplate Versioning Methods

CreateNewContractTemplateVersion

Description

This method creates a new version of the specified contract template.

Syntax

long CreateNewContractTemplateVersion (ContractTemplateData contractTemplateData, long contractTemplateFolderItemId);

Parameters

Name	Type	Description
contractTemplateData	ContractTemplateData	The data to use to create the new version.
ContractTemplateFolderItemId	long	The ID of the contract template folder item.

Return Value

The ID of the new contract template version, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ContractTemplateData object replaces the data of the existing contract template version, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding service components).

CommitContractTemplate

Description

This method commits the version changes of the specified contract template.

Syntax

ActionInfoCollection CommitContractTemplate (long contractTemplateFolderItemId, string Justification, bool forceCommit);

Parameters

Name	Type	Description
contractTemplateFolderItemId	long	The ID of the contract version folder item.
Justification	string	Text describing the reasons for committing the contract template changes, and any additional comments.
forceCommit	bool	Specifies whether to force the commit.

Return Value

An ActionInfoCollection object, which is a collection of zero or more ActionInfo items with information about the failure or success of the commit process.

Each ActionInfo item consists of:

- ActionInfoSeverity - An enumeration of Error, Warning, Info, or IneffectiveModulesMessages.
- Info - An information string.
- InfoCode - The system's identifier for the specific message.

Remarks

If the value of **forceCommit** is set to **false**, the contract is not committed at all; a notification will be issued if there is a Warning. If the **forceCommit** argument is set to **true**, the contract will be committed despite any warnings

ServiceDefinition Methods

ServiceDefinition Parameters Methods

GetServiceDefinitionParameters

Description

This method gets all contract level parameters for the specified service definition.

Syntax

ParametersCollection GetServiceDefinitionParameters (long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

The service definition parameters, as type ParametersCollection.

Remarks

None.

DeleteServiceDefinitionParameter

Description

This method deletes the specified contract level parameter of the specified service definition.

Syntax

void DeleteServiceDefinitionParameter (long serviceDefinitionFolderItemID, long ParameterID);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
ParameterID	long	The ID of the parameter to delete.

Return Value

None.

Remarks

None.

CreateServiceDefinitionParameter

Description

This method creates a contract level parameter for the specified service definition folder item.

Syntax

void CreateServiceDefinitionParameter (long serviceDefinitionFolderItemID, ParameterData parameterData);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained by invoking *GetServiceDefinitionParameters* (see page 251).

UpdateServiceDefinitionParameter

Description

This method updates a specific contract level parameter for the specified service definition.

Syntax

void UpdateServiceDefinitionParameter (long serviceDefinitionFolderItemID, ParameterData parameterData);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

Service Definition Methods

GetServiceDefinitionFolderItems

Description

This method gets a list of service level template folder items in which the name matches the search string.

Syntax

A typical syntax for this method is shown next.

EntityListData GetServiceDefinitionFolderItems(string searchText)

Parameters

	Description
	The text to search for. The search string is optional and can be empty. If used, it can contain an asterisk as a wildcard for the search.

Return Value

The list of the service level template folder items, as type EntityListData, includes the name and the folder item ID of service level templates.

Remarks

None.

GetServiceDefinitionFolderItemsByServiceID

Description

This method gets a list of service level template folder items in the system for the service contained as an included service.

Syntax

A typical syntax for this method is shown next.

NetityListData GetServiceDefinitionFolderItemsBlonyServiceID(long nServiceID)

Parameters

	Description
	The Service ID.

Return Value

The list of the service level template folder items, as type EntityListData which contains the name and the folder item ID of service level templates.

Remarks

None.

ServiceDefinition Details Methods

GetServiceDefinitionDetails

Description

This method gets all details for the specified service definition.

Syntax

ServiceDefinitionData GetServiceDefinitionDetails (long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

The service definition details, as type ServiceDefinitionData.

Remarks

The data returned is already filled with both the numerical IDs and the textual name of every sub-entity of which it consists.

UpdateServiceDefinitionDetails

Description

This method updates the details for the specified service definition.

Syntax

void UpdateServiceDefinitionDetails (ServiceDefinitionData serviceDefinitionData, long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
serviceDefinitionData	ServiceDefinitionData	The data to use to update the service definition.
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

None.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ServiceDefinitionData object replaces the data of the existing contract template, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding service components).

GetServiceDefinitionTimeSlots

Description

This method gets a list of all time slots of the specified service definition for which the name matches the search string.

Syntax

EntityListData GetServiceDefinitionTimeSlots (long serviceDefinitionFolderItemID, string searchText;

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The time slots, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetServiceDefinitionCustomAttributes

Description

This method gets all custom attributes for the specified service definition.

Syntax

CustomAttributesCollection GetServiceDefinitionCustomAttributes (long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

The custom attributes, as type CustomAttributesCollection.

Remarks

None.

ServiceDefinition Metrics Methods

GetServiceDefinitionMetrics

Description

This method gets a list of all metrics of the specified service definition for which the name matches the search string.

Syntax

EntityListData GetServiceDefinitionMetrics (long serviceDefinitionFolderItemID, string searchText);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The service definition metrics, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetServiceDefinitionLastCommittedMetrics

Description

This method returns a list of the metrics of the last committed version of the specified Service Definition for which the name matches the search string.

Syntax

EntityListData GetServiceDefinitionLastCommittedVersionMetrics (long serviceDefinitionFolderItemID, string searchText);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The metrics of the last, committed version of the service definition as type EntityListData.

Note: Returns metrics only if there is a committed version.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Note: The status (effective, not effective, pending) does not have an affect; the sole consideration is whether the template is “committed” or not committed.

Note: Throws an error when the service definition has no committed version.

DeleteServiceDefinitionMetric

Description

This method deletes the specified service definition metric.

Syntax

void DeleteServiceDefinitionMetric (long MetricId);

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.

Return Value

None.

Remarks

None.

CreateServiceDefinitionMetric

Description

This method creates a metric for the specified service definition.

Syntax

long CreateServiceDefinitionMetric (long serviceDefinitionFolderItemID, MetricData metricData);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
metricData	MetricData	The data to use to create the new metric.

Return Value

The ID of the new metric, as type long.

Remarks

None.

GetServiceDefinitionMetricDetails

Description

This method gets all details for the specified service definition metric.

Syntax

MetricData GetServiceDefinitionMetricDetails (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The metric details, as type MetricData.

Remarks

The data returned is already filled with both the numerical ID and the text name for every sub-entity that it contains.

UpdateServiceDefinitionMetricDetails

Description

This method updates metric details for the specified service definition.

Syntax

void UpdateServiceDefinitionMetricDetails (long serviceDefinitionFolderItemID, MetricData metricData);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
metricData	MetricData	The data to use to update the metric details.

Return Value

None.

Remarks

None.

DeleteServiceDefinitionMetricParameter

Description

This method deletes the specified parameter of the specified service definition metric.

Syntax

void DeleteServiceDefinitionMetricParameter (long MetricId, long ParameterId);

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.
ParameterId	long	The ID of the parameter.

Return Value

None.

Remarks

None.

UpdateServiceDefinitionMetricParameter

Description

This method updates a specific parameter of the specified service definition metric.

Syntax

void UpdateServiceDefinitionMetricParameter (long MetricId, ParameterData parameterData;

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.

Name	Type	Description
parameterData	ParameterData	The data to use to update the parameter.

Return Value

None.

Remarks

The ID of the parameter to update is contained in the ParameterData type class (see ParameterData Classes (see page 300)).

CreateServiceDefinitionMetricParameter

Description

This method creates a parameter for the specified metric.

Syntax

void CreateServiceDefinitionMetricParameter (long MetricId, ParameterData parameterData;

Parameters

Name	Type	Description
MetricId	long	The ID of the metric.
parameterData	ParameterData	The data to use to create the parameter.

Return Value

None.

Remarks

The ID of the newly created parameter can be obtained as part of the metric's parameters collection, in the metric's details, by invoking *GetServiceDefinitionMetricDetails* (see page 262).

AddMetricsToServiceDefinition

Description

This method copies existing metrics in the system to the specified service definition.

Syntax

void AddMetricsToServiceDefinition (long ServiceDefinitionFolderItemID, long[] MetricIDs, bool forceAdd);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemId	long	The ID of the service definition folder item.
MetricIDs	long[]	An array of IDs of the selected metrics to add.
forceAdd	bool	Specifies whether to force the addition.

Return Value

The copied metrics may belong to another contract, service definition, or contract template.

If the addition fails due to existing metrics with similar names, or mismatched effective dates, it is possible to set the **forceAdd** argument to **true**, in order to perform the addition while automatically renaming the metrics.

Remarks

None.

ServiceDefinition Metric Motes Methods

GetServiceDefinitionMetricNotes

Description

This method get the details of the metric notes in a service definition.

Syntax

MetricNotesData GetServiceDefinitionMetricNotes(long metricID);

Parameters

Name	Type	Description
metricID	long	The Metric's ID.

Return Value

The metric notes details, as type MetricNotesData.

Remarks

None.

UpdateServiceDefinitionMetricNotes

Description

This method updates the metric notes in a service definition.

Syntax

void UpdateServiceDefinitionMetricNotes(long metricID, MetricNotesData data);

Parameters

Name	Type	Description
metricID	long	The Metric's ID.
data	MetricNotesData	The data used to update notes of this metric.

Return Value

None.

Remarks

None.

ServiceDefinition Metric Registration Methods

AddServiceDefinitionMetricRegistration

Description

This method creates an event registration for the specified service definition metric.

Syntax

void AddServiceDefinitionMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to create the registration.

Return Value

None.

Remarks

The ID of the newly created registration can be obtained as part of the metric's registration collection, by invoking GetContractServiceDefinitionMetricRegistrations (see page 268).

UpdateServiceDefinitionMetricRegistration

Description

This method updates the specified event registration of the specified service definition metric.

Syntax

void UpdateServiceDefinitionMetricRegistration (long metricID, MetricRegistrationData metricRegistrationData);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
metricRegistrationData	MetricRegistrationData	The data to use to update the registration.

Return Value

None.

Remarks

The ID of the registration to update is contained in the MetricRegistrationData type class (see MetricRegistrationData (see page 296)).

DeleteServiceDefinitionMetricRegistration

Description

This method deletes the specified event registration of the specified service definition metric.

Syntax

void DeleteServiceDefinitionMetricRegistration (long metricID, long registrationID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.
registrationID	long	The ID of the registration.

Return Value

None.

Remarks

None.

GetServiceDefinitionMetricRegistrations

Description

This method gets all event registrations of the specified service definition metric.

Syntax

MetricRegistrationsCollection GetServiceDefinitionMetricRegistrations (long metricID);

Parameters

Name	Type	Description
metricID	long	The ID of the metric.

Return Value

The event registrations of the metric, as type MetricRegistrationsCollection.

Remarks

None.

ServiceDefinition Workflow Methods

ServiceDefinitionWorkflowRequestApproval

Description

This method sends an e-mail for an approval request as part of a service definition version's workflow.

The e-mail is automatically sent to the address that is set in the address field of the service definition's approver.

Syntax

void ServiceDefinitionWorkflowRequestApproval (long ServiceDefinitionFolderItemID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.
WorkFlowRemarks	string	Any remarks regarding the version and its approval request.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts preferences should be provided with the address of an accessible, valid mail server.

ServiceDefinitionWorkflowCancelApprovalRequest

Description

This method sends an e-mail cancelling a request for approval of a service definition version's workflow.

Syntax

void ServiceDefinitionWorkflowCancelApprovalRequest (long ServiceDefinitionFolderItemID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.
WorkFlowRemarks	string	Any remarks regarding the cancellation.

Return Value

None.

Remarks

In order to send e-mail messages, the system's alerts configuration should be provided with the address of an accessible, valid mail server.

ServiceDefinitionWorkflowApprove

Description

This method approves a service definition version's workflow.

Syntax

void ServiceDefinitionWorkflowApprove (long ServiceDefinitionFolderItemID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.
WorkFlowRemarks	string	Any remarks regarding the approval.

Return Value

None.

Remarks

None.

ServiceDefinitionWorkflowDeny

Description

This method denies a service definition version's workflow.

Syntax

void ServiceDefinitionWorkflowDeny (long ServiceDefinitionFolderItemID, string WorkFlowRemarks);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.
WorkFlowRemarks	string	Any remarks regarding the denial.

Return Value

None.

Remarks

None.

ServiceDefinition Detach/Delete Methods

DeleteServiceDefinition

Description

This method deletes the specified service definition.

Syntax

void DeleteServiceDefinition (long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

None.

Remarks

None.

DeleteServiceDefinitionLatestVersion

Description

This method deletes the latest version of the specified service definition.

Syntax

void DeleteServiceDefinitionLatestVersion (long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

None.

Remarks

None.

ServiceDefinition Versioning Methods

CreateNewServiceDefinitionVersion

Description

This method creates a new version of the specified service definition.

Syntax

long CreateNewServiceDefinitionVersion (ServiceDefinitionData serviceDefinitionData, long serviceDefinitionFolderItemID);

Parameters

Name	Type	Description
serviceDefinitionData	ServiceDefinitionData	The data to use to create the new version.
ServiceDefinitionFolderItemID	long	The ID of the service definition folder item.

Return Value

The ID of the new service definition version, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The data provided in the ServiceDefinitionData object replaces the data of the existing contract template version, and cannot be left unfilled or partially filled. An ID for every sub-entity of which the data structure consists must be provided (for example, the contract's corresponding service components).

CommitServiceDefinition

Description

This method commits the specified service definition for approval.

Syntax

ActionInfoCollection CommitServiceDefinition (long serviceDefinitionFolderItemID, string Justification, bool forceCommit);

Parameters

Name	Type	Description
serviceDefinitionFolderItemID	long	The ID of the service definition folder item.
Justification	string	Text describing the reasons for committing the service definition changes, and any additional comments.
forceCommit	bool	Specifies whether to force the commit.

Return Value

An ActionInfoCollection object, which is a collection of zero or more ActionInfo items with information about the failure or success of the commit process.

Each ActionInfo item consists of:

- ActionInfoSeverity - An enumeration of **Error**, **Warning**, **Info**, or **IneffectiveModulesMessages**.
- Info - An information string.
- InfoCode - The system's identifier for the specific message.

Remarks

If the value of **forceCommit** is set to **false**, the contract is not committed at all; a notification will be issued if there is a Warning. If the **forceCommit** argument is set to **true**, the contract will be committed despite any warnings

ServiceDefinition Creation Methods

CreateServiceDefinition

Description

This method creates a service definition from the input data.

Syntax

long CreateServiceDefinition (ServiceDefinitionData serviceDefinitionData, PortfolioContainerType ServiceDefinitionContainerType, long ServiceDefinitionContainerID);

Parameters

Name	Type	Description
serviceDefinitionData	ServiceDefinitionData	The data to use to create the new service definition.
ServiceDefinitionContainerType	PortfolioContainerType	The container type of the new service definition.
ServiceDefinitionContainerID	long	The container ID of the new service definition.

Return Value

The ID of the new service definition, as type long.

Remarks

The method validates the data sent to it and may throw a validation exception upon failure.

The ServiceDefinitionData object provided for this method should be filled with an ID for every sub-entity of which it consists (for example, the contract's corresponding primary contract party).

Repository Interface

The Repository Interface methods are listed in the following tables.

Service Methods

GetCurrencies (see page 277)	GetDomainCategories (see page 287)
CreateServiceComponent (see page 284)	GetDomainCategoryDetails (see page 288)
GetServiceComponents (see page 285)	GetTimeSlotTemplates (see page 289)
CreateServiceDomain (see page 285)	GetTimeZones (see page 290)

GetServiceDomains (see page 286)	CreateUnit (see page 290)
CreateDomainCategory (see page 277)	GetUnits (see page 291)

Contract Party Methods

CreateContractParty (see page 277)	GetContractPartyDetails (see page 279)
DeleteContractParty (see page 278)	UpdateContractParty (see page 279)
GetContractPartyDetails	

Customer Attribute Methods

GetCustomAttributes (see page 280)	GetCustomAttributeDetails (see page 281)
UpdateCustomAttribute (see page 281)	

Service Component Methods

CreateServiceComponent (see page 284)	
GetServiceComponents (see page 285)	UpdateServiceComponentDetails (see page 283)
GetServiceComponentDetails (see page 282)	DeleteServiceComponent (see page 283)

TimeSlotTemplate Methods

GetTimeSlotTemplates (see page 289)

TimeZone Methods

GetTimeZones (see page 290)

Currency Methods

GetCurrencies

Description

This method gets a list of all currencies in the system for which the name matches the search string.

Syntax

EntityListData GetCurrencies (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The currencies, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

ContractPartyMethods

CreateContractParty

Description

This method creates a contract party from the input data.

Syntax

long CreateContractParty(ContractPartyData ContractPartyData);

Parameters

Name	Type	Description
ContractPartyData	ContractPartyData	The data used to create the Contract Party.

Return Value

The ID of the new contract party, as type long.

Remarks

None.

DeleteContractParty

Description

This method deletes the specified contract party.

Syntax

void DeleteContractParty(long ContractPartyId);

Parameters

Name	Type	Description
ContractPartyID	long	The ID of the Contract Party

Return Value

None.

Remarks

None.

GetContractPartyDetails

Description

This method gets the details of the specified contract party.

Syntax

ContractPartyData GetContractPartyDetails(long ContractPartyId);

Parameters

Name	Type	Description
ContractPartyID	long	The Contract Party ID.

Return Value

The contract party details, as type ContractPartyData.

Remarks

None.

UpdateContractParty

Description

This method updates all contract party details

Syntax

void UpdateContractParty(ContractPartyData ContractPartyData);

Parameters

Name	Type	Description
ContractPartyData	ContractPartyData	The data used to update the contract party.

Return Value

None.

Remarks

The contract party which is updated is specified by the ID field of ContractPartyData.

Customer Attribute Methods

GetCustomAttributes

Description

This method gets a list of all customer attributes in the system for which the name matches the search string.

Syntax

EntityListData GetCustomAttributes(string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and can be left empty. If used, it can contain an asterisk as a wildcard for the search.

Return Value

The details of the customer attributes, as type CustomAttributeDefinitionData.

Remarks

None.

GetCustomAttributeDetails

Description

This method gets the details of the specified customer attribute.

Syntax

CustomAttributeDefinitionData GetCustomAttributeDetails(long customAttributeId);

Parameters

Name	Type	Description
customAttributeID	long	The ID of the customer attributes.

Return Value

The details of the customer attributes, as type CustomAttributeDefinitionData.

Remarks

None.

UpdateCustomAttribute

Description

This method updates the details of the specified customer attribute.

Syntax

void UpdateCustomAttribute(CustomAttributeDefinitionData customAttributeDefinitionData);

Parameters

Name	Type	Description
customAttributeDefinitionData	CustomAttributeDefinitionData	The ID of the customer attributes.

Return Value

The data used to update the contract customer attribute.

Remarks

The customer attribute which is updated is specified by the filed ID of CustomAttributeDefinitionData.

Service Component Methods

ServiceComponentDetails

Description

This method gets the details of the specified service component.

Syntax

ServiceComponentData GetServiceComponentDetails(long serviceComponentID);

Parameters

Name	Type	Description
serviceComponentID	long	The ID of the service component.

Return Value

The details of service component, as type ServiceComponentData.

Remarks

None.

DeleteServiceComponent

Description

This method deletes the specified service component.

Syntax

void DeleteServiceComponent(long serviceComponentID);

Parameters

Name	Type	Description
serviceComponentID	long	The ID of the service component.

Return Value

None.

Remarks

None.

UpdateServiceComponentDetails

Description

This method updates the details of the specified service component.

Syntax

void UpdateServiceComponentDetails(ServiceComponentData serviceComponentData);

Parameters

Name	Type	Description
serviceComponentData	serviceComponentData	The data used to update the service component.

Return Value

None.

Remarks

The service component which is updated is specified by the field id of ServiceComponentData.

ServiceComponent Methods

CreateServiceComponent

Description

This method creates a service component from the input data.

Syntax

long CreateServiceComponent (ServiceComponentData serviceComponentData);

Parameters

Name	Type	Description
serviceComponentData	ServiceComponentData	The data to use to create the service component.

Return Value

The ID of the new service component, as type long.

Remarks

None.

GetServiceComponents

Description

This method gets a list of all service components in the system for which the name matches the search string.

Syntax

EntityListData GetServiceComponents (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The service components, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

ServiceDomain Methods

CreateServiceDomain

Description

This method creates a domain from the input data.

Syntax

long CreateServiceDomain (ServiceDomainData serviceDomainData);

Parameters

Name	Type	Description
serviceDomainData	ServiceDomainData	The data to use to create the service domain.

Return Value

The ID of the new service domain, as type long.

Remarks

None.

GetServiceDomains

Description

This method gets a list of all service domains in the system for which the name matches the search string.

Syntax

EntityListData GetServiceDomains (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The service domains, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

DomainCategory Methods

CreateDomainCategory

Description

This method creates a domain category from the input data.

Syntax

long CreateDomainCategory (DomainCategoryData domainCategoryData);

Parameters

Name	Type	Description
domainCategoryData	DomainCategoryData	The data to use to create the domain category.

Return Value

The ID of the new domain category, as type long.

Remarks

None.

GetDomainCategories

Description

This method gets a list of all domains categories in the system for which the name matches the search string.

Syntax

EntityListData GetDomainCategories (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The domain categories, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

GetDomainCategoryDetails

Description

This method gets all details for the specified domain category.

Syntax

DomainCategoryData GetDomainCategoryDetails(long domainCategoryId);

Parameters

Name	Type	Description
domainCategoryId	long	The ID of the domain category.

Return Value

The domain category details, as type DomainCategoryData.

Remarks

None.

TimeSlotTemplate Methods

GetTimeSlotTemplates

Description

This method gets a list of all time slot templates in the system for which the name matches the search string.

Syntax

EntityListData GetTimeSlotTemplates (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The time slot templates, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

TimeZone Methods

GetTimeZones

Description

This method gets a list of all time zones in the system for which the name matches the search string.

Syntax

EntityListData GetTimeZones (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The time zones, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Unit Methods

CreateUnit

Description

This method creates a unit from the input data.

Syntax

long CreateUnit (UnitData unitData);

Parameters

Name	Type	Description
unitData	UnitData	The data to use to create the unit.

Return Value

The ID of the new unit, as type long.

Remarks

None.

GetUnits

Description

This method gets a list of all units (for metric measurement) in the system for which the name matches the search string.

Syntax

EntityListData GetUnits (string searchText);

Parameters

Name	Type	Description
searchText	string	The text to search for. The search string is optional and may be left empty. If used, it may contain an asterisk as a wildcard for the search.

Return Value

The units, as type EntityListData.

Remarks

An EntityListData object is returned as a collection of EntityListItemData items, which contains both the name and ID of every object.

Open API Object Classes

The Open API object classes can be divided into four parts:

- Common Global Data Classes (see page 292)
- Contract Classes (see page 302)
- Portfolio Classes (see page 304)
- Repository Classes (see page 307)

Common Global Data Classes

The Common Global Data classes are listed in the following table.

ContractServiceComponentItemData (see page 292)	MetricRegistrationData (see page 296)
ContractWorkflowData (see page 293)	MetricGeneralDetailsSection (see page 298)
EntityReferenceData (see page 293)	MetricGranularitySection (see page 299)
EntityListItemData (see page 294)	ParameterReferenceData (see page 300)
ActionInfo	ParameterData (see page 301)
MetricData (see page 295)	

Contract Related Common Classes

ContractServiceComponentItemData

Description

This class is used to represent the service component item data of contracts, contract templates, and service definitions.

Members

Name	Type	Description
ContractServiceComponent	EntityReferenceData	Reference by ID to the contract service component.
Fee	long	The contract or service fee.
Seats	long	The number of seats for a contract or service.

Remarks

None.

ContractWorkflowData

Description

This class is used to represent the workflow data of contracts, contract templates, and service definitions.

Members

Name	Type	Description
WorkflowEnabled	bool	Specifies whether the workflow is enabled.
WorkflowApproverEmailAddress	string	The e-mail address of the workflow approver.
WorkflowReplyToEmailAddress	string	The e-mail address to which the workflow approver should reply.

Remarks

The workflow approver is the person who has been requested to approve the workflow. This person can either approve, or deny approval of, the workflow.

When a workflow method is invoked (*ContractWorkflowRequestApproval* (see page 220)), an email is automatically sent to the address specified in the **WorkflowApproverEmailAddress** field, provided that:

- The **WorkflowEnabled** field is set to **true**.
- The system's alerts configuration is set to send e-mails.

Basic Data Classes

EntityReferenceData

Description

This class is used to represent the data of entity references.

Members

Name	Type	Description
EntityID	long	The ID of the entity. This field is validated. It should contain a valid ID, which is number greater than 0.
EntityName	string	The name of the entity.

Remarks

None.

EntityListItemData

Description

This class is used to represent an item in a list of entities of any type, returned by some of the Open Interface's methods. Every item on the list consists of an ID and a name of the entity.

Members

Name	Type	Description
EntityID	long	The ID of the entity.
EntityName	string	The name of the entity.

Remarks

None.

ActionInfo

Description

This class is used to represent the information that is returned from the system when certain methods are invoked (for example, CommitContract, which returns a collection of ActionInfo items).

Members

Name	Type	Description
severityLevel	ActionInfoSeverity	The severity level of the action. An enumeration of: ■ **Error** ■ **Warning** ■ **Info** ■ **IneffectiveModulesMessages**
Info	string	A descriptive information string.
InfoCode	string	The system's identifier for the specific message.

Remarks

None.

OptionalDoubleEntity

Description

This class is used to represent data that contains a value of null or of type double.

Members

Name	Type	Description
IsValueLess	Boolean	Specifies whether or not a value exists.
Value	double	Specifies the value.

Remarks

Dates from version 7.0 SP1.

MetricData Classes

MetricData

Description

This class is used to represent metric data.

Members

Name	Type	Description
MetricType	EntityReferenceData	Reference by ID to the metric type. The value may not be null.
MetricSection	EntityReferenceData	Reference by ID to the metric section. One of: ■ **Contractual** ■ **Financial** ■ **Operational** The value may not be null.

Name	Type	Description
ID	long	The ID of the metric.
Name	string	The name of the metric.
Description	string	The description of the metric.
generalDetails	MetricGeneralDetailsSection	The general details of the metric.
parametersData	ParametersCollection	The parameter data of the metric.
granularityDetails	MetricGranularitySection	The granularity details of the metric.

Remarks

When using the CreateContractMetric method the following fields are set to their default values:

1. Objective Statement
2. Clustering
3. Business Logic

However, the structure itself does **not** contain these fields.

Note: The above three fields exist in the MetricData method itself however, these particular fields are **not** accessible to the user/customer via this method. Therefore, for example - the Objective Statement is visible in the GUI but cannot be changed by means of this method.

MetricRegistrationData

Description

This class is used to represent data of metric registrations.

Members

Name	Type	Description
RegistrationID	long	The ID of the registration.
RegistrationType	RegistrationType	An enumeration of the type of the registration. See Remarks.
Description	string	The description of the registration.
BusinessLogicMethod	string	The name of the business logic method that handles events by this registration.

Name	Type	Description
EventType	EntityReferenceData	Reference by ID to the registered event type (if applicable for the registration type).
Resource	EntityReferenceData	Reference by ID to the registered resource (if applicable for the registration type).
ResourceType	EntityReferenceData	Reference by ID to the registered resource type (if applicable for the registration type).
ResourceGroup	EntityReferenceData	Reference by ID to the registered resource group (if applicable for the registration type).
SourceMetric	EntityReferenceData	Reference by ID to the registered source metric (if applicable for the registration type).
WithContractParty	bool	Specifies whether the registration includes the contract's primary contract party.
WithServiceComponent	bool	Specifies whether the registration includes the metric's service component.
RegistrationTimePeriod	TimePeriod	An enumeration of the type of the registration time period.

Remarks

RegistrationTypes are as follows:

ContractParty,
Service,
ContractPartyAndService,
ResourceGroup,
Resource,
Advanced,
Metric,
ClusterItem

MetricGeneralDetailsSection

Description

This class is used to represent the data of the general details of metrics.

Members

Name	Type	Description
MainIndicatorType	EntityReferenceData	Reference by ID to the main indicator of the metric.
ServiceComponent	EntityReferenceData	Reference by ID to the service component of the metric.
ServiceDomain	EntityReferenceData	Reference by ID to the service domain of the metric.
DomainCategory	EntityReferenceData	Reference by ID to the domain category of the metric.
Unit	EntityReferenceData	Reference by ID to the measurement unit of the metric.
Timeslot	EntityReferenceData	Reference by ID to the time slot of the metric.
TimeZone	EntityReferenceData	Reference by ID to the time zone of the metric.
IsTargetLess	bool	Specifies whether target exists or not.
Target	double	The target of the metric.
DynamicTarget	bool	Specifies whether the metric target is dynamic.
TrackingPeriod	long	The tracking period of the metric.
Currency	EntityReferenceData	Reference by ID to the currency of the metric.
MeasurabilityStatus	MeasurabilityStatus. Statuses	An enumeration representing the measurability status of the metric.
UnitOfConsumption	EntityReferenceData	Reference by ID to the unit of consumption of the metric.
Forcasted	bool	Specifies whether the metric is forecasted.
ForcastTable	string	The forecast table of the metric. Represented by an XML string.

Name	Type	Description
PricePerUnitTable	string	The price-per-unit table of the metric. Represented by an XML string.

Remarks

The field "IsTargetLess" dates from version 7.0 SP1. The field "Target" was changed from long to double as of version 7.0 SP1.

MetricGranularitySection

Description

This class is used to represent the granularity data of metrics.

Members

Name	Type	Description
CalculateYear	bool	Determines whether the metric is to be calculated once a year.
CalculateQuarter	bool	Determines whether the metric is to be calculated once every three months (every quarter year).
CalculateMonth	bool	Determines whether the metric is to be calculated once a month.
CalculateWeek	bool	Determines whether the metric is to be calculated once a week.
CalculateDay	bool	Determines whether the metric is to be calculated once a day.
CalculateHour	bool	Determines whether the metric is to be calculated once an hour.

Remarks

None.

MetricNotes Classes

The MetricNotes classes are listed in the following table.

MetricNotes (see page 300)

MetricNotes

Description

This class is used to represent data for metric notes.

Members

Name	Type	Description
Notes	string	The string content of DOCX or HTML for metric notes.
NoteType	string	The type of metric notes. One of the following: ■ DOCX ■ HTML

Remarks

None.

ParameterData Classes

ParameterReferenceData

Description

This class is used to represent the data of parameter references.

Members

Name	Type	Description
RefParameterID	long	The ID of the parameter reference.
RefParameterTableIndex RowValueInSelectedColumn	string	The row value in the selected column of the parameter reference table index. Relevant if referenced parameter is of type **table**.
RefParameterTableColumnName	string	The table column name of the parameter reference. Relevant if referenced parameter is of type **table**.

Remarks

None.

ParameterData

Description

This class is used to represent parameter data.

Members

Name	Type	Description
ParameterName	string	The name of the parameter.
parameterID	long	The ID of the parameter.
ParameterDescription	string	The description of the parameter.
parameterType	ParameterType	An enumeration of the type of the parameter.
ParameterValue	string	The value of the parameter, according to its type: ■ **number** ■ **text** ■ **date** ■ **table** (represented by legal XML string) ■ **list** (represented by a string that contains the default value) ■ **hyperlink** (contains the link's URL)
ParameterListValues	String[]	Array of strings for the list legal/optional values. Relevant only if parameter is of type **list**.
ParameterHyperLinkText	string	The hyperlink text of the parameter. Relevant only if parameter is of type **hyperlink**.

Name	Type	Description
ParameterValidation	string	Optional. The validation of the parameter. Relevant only if parameter is of type **number**. It is an XML string representing the number's minimum, maximum, and decimal precision restrictions. The parameter's value must conform to these restrictions.
RefParamData	ParameterReferenceData	The parameter reference data.
ParameterOrder	long	The order parameter.

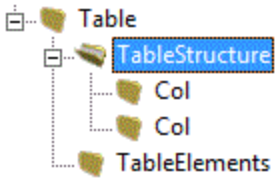
Remarks

The field "ParameterOrder" dates from version 7.0 SP1.

An example of data in a properly formatted, legal XML type table is given below:

```
<Table>
<TableStructure>
<Col Name="Kunde" Index="Y" Mandatory="N" Type="Text" />
<Col Name="Anteil_in_Prozent" Index="N" Mandatory="N" Type="Number" />
</TableStructure>


<TableElements />
</Table>
```



Contract Classes

The Contract classes are listed in the following table.
ContractData (see page 303)

ContractData Classes

ContractData

Description

This class is used to represent contract data.

Members

Name	Type	Description
ContractName	string	The contract name. The value may not be null.
IncludedService Components	ContractService ComponentCollecti on	The service components that are included. The value may not be null.
ExcludedService Components	ContractService ComponentCollecti on	The service components that are excluded. The value may not be null.
ValidFrom	DateTime	The start date and time (from which the contract is valid). The value may not be null.
ValidTo	DateTime	The end date and time (until which the contract is valid). The value may not be null.
ContractTimeZone	EntityReferenceDat a	Reference by ID to the time zone for the contract. The value may not be null.
ContractWorkflow	ContractWorkflowD ata	The workflow data for the contract.
CustomAttributes	CustomAttributes Collection	The custom attributes for the contract.
Description	string	The description of the service definition. Maximum string length is 512 characters.
ContractType	EntityReferenceDat a	Reference by ID to the contract type. The value may not be null.
Currency	EntityReferenceDat a	The currency to use. The value may not be null.

Name	Type	Description
Fee	long	The fee.
Seats	long	The number of seats.
Owner	string	The owner. Maximum string length is 60 characters.
PrimaryContractParty	EntityReferenceData	Reference by ID to the primary contract party. The value may not be null.
PrimaryContractPartyIsProvider	bool	Specifies whether the primary contract party is the service provider.
SecondaryContractParty	EntityReferenceData	Reference by ID to the secondary contract party.

Remarks

The **ExcludedServiceComponents** field may not be null, but it may contain a list with no items. It is used for updating existing contracts, to detach service components from them.

An automatic validation is performed to verify that **ValidTo** is later than **ValidFrom**.

Portfolio Classes

The Portfolio classes are listed in the following table.

[ContractTemplateData](#) (see page 304) [ServiceDefinitionData](#) (see page 306)

[PortfolioEntity](#) (see page 306)

ContractTemplateData Classes

ContractTemplateData

Description

This class is used to represent contract template data.

Members

Name	Type	Description
ContractName	string	The contract name. The value may not be null.

Name	Type	Description
IncludedServiceComponents	ContractServiceComponentCollection	The service components that are included. The value may not be null.
ExcludedServiceComponents	ContractServiceComponentCollection	The service components that are excluded. The value may not be null.
ValidFrom	DateTime	The start date and time (from which the contract is valid). The value may not be null.
ValidTo	DateTime	The end date and time (until which the contract is valid). The value may not be null.
ContractTimeZone	EntityReferenceData	Reference by ID to the time zone for the contract. The value may not be null.
ContractWorkflow	ContractWorkflowData	The workflow data for the contract.
CustomAttributes	CustomAttributesCollection	The custom attributes for the contract.
Description	string	The description of the service definition. Maximum string length is 512 characters.
ContractType	EntityReferenceData	Reference by ID to the contract type. The value may not be null.
Currency	EntityReferenceData	The currency to use. The value may not be null.
Fee	long	The fee.
Seats	long	The number of seats.
Owner	string	The owner. Maximum string length is 60 characters.

Remarks

The **ExcludedServiceComponents** item may not be null, but it may contain a list with no items. It is used for updating existing service definitions, to detach service components from them.

An automatic validation is performed to verify that **ValidTo** is later than **ValidFrom**.

PortfolioData Classes

PortfolioEntity

Description

This class is used to represent data for portfolio entries entries in entities' lists, when exploring the portfolio tree through the *GetPortfolioItems* (see page 226) and *GetCatalogItems* (see page 226) methods.

Members

Name	Type	Description
EntityType	PortfolioEntityType	The entity type of the portfolio.
EntityId	long	The ID of the portfolio.
EntityName	string	The name of the portfolio.

Remarks

None.

ServiceDefinitionData Classes

ServiceDefinitionData

Description

This class is used to represent data for service definitions.

Members

Name	Type	Description
ContractName	string	The contract name. The value may not be null.
IncludedServiceComponents	ContractServiceComponentCollection	The service components that are included. The value may not be null.
ExcludedServiceComponents	ContractServiceComponentCollection	The service components that are excluded. The value may not be null.
ValidFrom	DateTime	The start date and time (from which the contract is valid). The value may not be null.

Name	Type	Description
ValidTo	DateTime	The end date and time (until which the contract is valid). The value may not be null.
ContractTimeZone	EntityReferenceData	Reference by ID to the time zone for the contract. The value may not be null.
ContractWorkflow	ContractWorkflowData	The workflow data for the contract.
CustomAttributes	CustomAttributesCollection	The custom attributes for the contract.
Description	string	The description of the service definition. Maximum string length is 512 characters.

Remarks

The **ExcludedServiceComponents** item may not be null, but it may contain a list with no items. It is used for updating existing service definitions, to detach service components from them.

An automatic validation is performed to verify that **ValidTo** is later than **ValidFrom**.

Repository Classes

The Repository classes are listed in the following table.

ContractPartyData (see page 308)	DomainCategoryData (see page 312)
CustomAttributeData (see page 309)	ServiceComponentData (see page 313)
CustomAttrributeAttachment (see page 310)	ServiceDomainData (see page 314)
CustomAttributeDefinitionData (see page 311)	UnitData (see page 314)

ContractPartyData Classes

The ContractPartyData classes are listed in the following table.

ContractPartyData (see page 308)

ContractPartyData

Description

This class is used to represent data for contract party.

Members

Name	Type	Description
ID	long	The contract party ID.
Name	string	The contract party name (cannot be null).
Type	string	The type of contract party. The type must be one of the following types: -Provider -Customer -Internal Provider -Internal Receiver
Description	string	The contract party description.
Address	string	The address of the contract party.
Country	string	The country of the contract party .
State	string	The state of the contract party. Its numerical value must be larger than 0.
ZipCode	string	The zip code of the contract party.
Contact	string	The contact of the contract party.
PhoneNumb er1	string	The first phone number of the contract party.
PhoneNumb er2	string	The second phone number of the contract party.
FaxNumber	string	The contact party fax number.
MailAddress	string	The contact contract mail address.
Notes	string	Notes for the contract party.
Class	string	The contact contract party class.
Seats	long	The contact contract party seats.
Registration Date	DateTime	The registration date time of the contract party.

Name	Type	Description
ContractPartyGroups	AttachedContractPartyGroupCollection	The contract party attachments, As type AttachedEntityCollection<AttachedContractPartyGroup>

Remarks

None.

CustomAttributeData Classes

CustomAttributeData

Description

This class is used to represent data for custom attributes.

Members

Name	Type	Description
CustomAttributeDefinitionID	long	The ID of the custom attribute.
Name	string	The name of the custom attribute.
Type	CustomAttributeValueType	Enumeration of the type of the custom attribute. One of the following: ■ Text ■ Number ■ List ■ Hyperlink
IsMandatory	bool	Specifies whether or not the custom attribute must be set with an appropriate value. If the custom attribute does not have to be set, it can be omitted.
Value	string	The value of the custom attribute, according to its type.
HyperLinkDisplayText	string	The hyperlink display text of the custom attribute. Relevant only when the custom attribute is of type **hyperlink**, in which case the **Value** field contains the link's URL and this field contains the link's display text.

Name	Type	Description
ListValues	List<string>	The list values of the custom attribute. Relevant only when the custom attribute is of type **list**. Instead of containing a single string to represent the custom attribute's value, this field contains a list of strings representing the list's values.

Remarks

None.

CustomAttributeDefinitionData Classes

The CustomAttributeDefinitionData classes are listed in the following table.

CustomAttributeAttachment (see page 310)	CustomAttributeDefinitionData (see page 311)

CustomAttributeAttachment

Description

This class is used to represent an attachment data for custom attributes in definition.

Members

Name	Type	Description
ID	long	The custom attribute ID.
EntityType	CustomAttributeAttachmentEntityType	Enumeration of the value type of the custom attribute (must be one of the following): ■ AllEntities ■ ResourceType ■ MetricType ■ ResourceGroupType
AllEntityType	EntityReferenceData	Reference by ID to the AllEntities type.
ResourceType	EntityReferenceData	Reference by ID to the ResourceType type.
MetricType	EntityReferenceData	Reference by ID to the MetricType type.
IsMandatory	bool	The attachment is mandatory or not mandatory.

Name	Type	Description
DefaultValue	string	The default value for attachment of the attribute.

Remarks

The DefaultValue is in text even for non-text attributes' attachment.

CustomAttributeDefinitionData

Description

This class is used to represent data for contract party.

Members

Name	Type	Description
ID	long	The custom attribute ID.
Name	string	The custom attribute name.
Description	string	The custom attribute description.
Type	long	The attribute type.
ValueType	CustomAttributeValue Type	Enumeration of the value type of the custom attribute (must be one of the following): ■ Text ■ Number ■ List ■ Hyperlink
DefaultValue	string	The default value of the attribute.
IsExternalDataSourceEnabled	bool	Enable the external data source or not.
ListValues	CustomAttributeListValueCollection	The list of string for the attribute.
AttachmentsData	CustomAttributeListValueCollection.	CustomAttributeAttachment list of the attribute attachments.

Remarks

None.

DomainCategoryData Classes

DomainCategoryData

Description

This class is used to represent domain category data.

Members

Name	Type	Description
ID	long	Domain Category ID.
Name	string	The name of the domain category.
Description	string	The description of the domain category.
DefaultUnitOf Measurement	EntityReferenceDa ta	Reference by ID to the default unit by which the domain category is measured.
ServiceDomain	EntityReferenceDa ta	Reference to the ID of the ServiceDomain to which the DomainCategory belongs.
DefaultService ComponentLevelTarg et	TargetLimitType	An enumeration of the default service component level target (min/max).
MinimalService ComponentLevelTarg et	OptionalDoubleEnt ity	The minimal service component level target for the domain category.
MaximalService ComponentLevelTarg et	OptionalDoubleEnt ity	The maximal service component level target for the domain category.
Precision	long	The report precision for the domain category.
IsPredictionMetricActi ve	bool	Specifies whether the prediction metric is active for the domain category.
PredictionMetric OptimisticValue	double	The prediction metric optimistic value.
PredictionMetric PessimisticValue	double	The prediction metric pessimistic value.
PredictionMetric	EntityReferenceDa ta	Reference by ID to the prediction metric.
ThresholdsReferTo	ThresholdsReferen ce	An enumeration of the threshold reference (refers to either the deviation or the measurement).

Name	Type	Description
YellowThreshold	OptionalDoubleEntity	Value for the yellow threshold.
OrangeThreshold	OptionalDoubleEntity	Value for the orange threshold.
RedThreshold	OptionalDoubleEntity	Value for the red threshold.

Remarks

The following fields were changed from type long to type OptionalDoubleEntity as of version 7.0 Service Pack 1:

YellowThreshold, OrangeThreshold, RedThreshold, MinimalServiceComponentLevelTarget, MaximalServiceComponentLevelTarget.

In addition, the fields "ServiceDomain" and "ID" were added as of version 7.0 Service Pack 1.

ServiceComponentData Classes

The CustomAttributeDefinitionData classes are listed in the following table.
ServiceComponentData

ServiceComponentData

Description

This class is used to represent data for service components.

Members

Name	Type	Description
sServiceComponentName	string	The name of the service component.
sServiceComponentDescription	string	The description of the service component.
iServiceComponentFee	int	The fee for the service component.
IServiceComponentID	long	The service component ID.
bIs_Default	bool	Should always be set to **false**.

Name	Type	Description
bIs_Fee_Seats_Required	bool	When the user attaches this ServiceComponent to a contract, contract template, or service definition, this field specifies whether the user is also required to specify fee and seats .

Remarks

None.

ServiceDomainData Classes

ServiceDomainData

Description

This class is used to represent service domain data.

Members

Name	Type	Description
Name	string	The name of the service domain.
Description	string	The description of the service domain.

Remarks

None.

UnitData Classes

UnitData

Description

This class is used to represent service domain data.

Members

Name	Type	Description
Name	string	The name of the unit.
Description	string	The description of the unit.
Symbol	string	The textual symbol of the unit.

Name	Type	Description
StepSize	long	The step size of the unit.
UnitType	EntityReferenceData	Reference by ID to the unit's type.

Remarks

None.

Chapter 9: Launcher Page

This section contains the following topics:

Overview

Custom attributes and parameter are <isn> entities. They both represent a common data entry related to a number of other <isn> entities, such as: Contracts, Service Definitions, Resources, etc. This chapter describes the Launcher Page. Launcher Pager, in this sense, refers to having an extra button to the right of a customizable entity. Upon clicking this button, a modal dialog opens, displaying a customer crafted page that eases the value selection for that entity.

The launch page consists of an HTML page with an iFrame, pointing to a page from any URL declared in the CA Business Service Insight configuration. This enables displaying an external page with deployment-specific data (e.g. user list from Active Directory).

CA Business Service Insight passes a JavaScript object as an argument to the launch page. This object is available to the iFramed page. After calling the server, this external page updates the JavaScript object as needed, passes it to the launch page and sends it back to CA Business Service Insight for control update.

The figure below indicates a component diagram of the Launcher Page.

JavaScript API

JavaScript Objects

Entity Transferred

This is the entity that represents the custom attribute or parameter CA Business Service Insight entity. It contains all relevant data about these CA Business Service Insight entities and the context data that hosts these CA Business Service Insight entities.

The entity transferred information is:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  This is the name of the custom attribute or parameter CA Business Service Insight entity from which the page was launched.

- Type

  Can be custom attribute or parameter.

- ValueType

  This is the type of what the CA Business Service Insight entity can be (e.g. number, text, list, hyperlink or table).

- Value

  The current value that will be inserted into the CA Business Service Insight entity.

- IsMandatory

  Is this CA Business Service Insight entity mandatory in the hosting entity. Can be true or false.

- ContextData

  Contains objects that hold information about the hosting entity, which can be:

  - Resource
  - Resource Group
  - Contract
  - Contract Template
  - Service Definition
  - Metric

  ContextData also contains the UserContextData object.

Resource

The Resource object contains the information about the resource that hosts the custom attribute CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  The resource's name.

- Type

  Always Resource.

- EffectiveStartDate

  The date the resource is effective from.

- IsEffective

  Is the resource effective or not. Can be Yes or No.

- ResourceTypesName

  An array of strings of all resource types for the current resource.

- ContractParties

  An array of strings of all contract parties for the current resource.

- ServiceComponents

  An array of strings of all service components for the current resource.

- CustomAttributes

  An array of objects of all custom attributes that are defined for the current resource. The type of these objects is Custom Attribute (e.g. Entity Transferred).

- ResourceGroups

  An array of strings of all resource groups for the current resource.

Resource Group

The Resource Group object contains the information about the resource group that hosts the custom attribute CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  The resource group's name.

- Type

  Always Resource group.

- EffectiveStartDate

  The date the resource group is effective from.

- IsEffective

  Is the resource group effective or not. Can be Yes or No.

- ResourceTypesName

  An array of strings of all resource types for the current resource group.

- ContractParties

  An array of strings of all contract parties for the current resource group.

- ServiceComponents

  An array of strings of all service components for the current resource group.

- CustomAttributes

  An array of objects of all custom attributes that are defined for the current resource group. The type of these objects is Custom Attribute (e.g. Entity Transferred).

Contract

The Contract object contains the information about the contract that hosts the Contract parameter CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  The contract's name.

- Type

  Always Contract.

- EffectiveStartDate

  The date the contract is effective from.

- EffectiveEndDate

  The date the contract is effective to.

- ContractVersionID

  The current version ID of the contract.

- VersionValidFromDate

  The date this version of the contract is valid from.

- VersionValidToDate

  The date this version of the contract is valid to.

- ContractType

  The abbreviated type of the contract (e.g. SLA).

- ContractTypeName

  The full name of the contract's type (e.g. Service Level Agreement)

- ContractPartyReceiverName

  The contract's contract party receiver name.

- ContractPartyProviderName

  The contract's contract party provider name.

Contract Template

The Contract Template object contains the information about the contract template that hosts the Contract Template parameter CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  The contract template's name.

- Type

  Always Contract Template.

- EffectiveStartDate

  The date the contract template is effective from.

- EffectiveEndDate

  The date the contract template is effective to.

- ContractVersionID

  The current version ID of the contract template.

- VersionValidFromDate

  The date this version of the contract template is valid from.

- VersionValidToDate

  The date this version of the contract template is valid to.

- ContractType

  The abbreviated type of the contract template (e.g. SLA).

- ContractTypeName

  The full name of the contract template's type (e.g. Service Level Agreement)

Service Definition

The Service Definition object contains the information about the service definition that hosts the Service Definition parameter CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  The service definition's name.

- Type

  Always Service Definition.

- EffectiveStartDate

  The date the service definition is effective from.

- EffectiveEndDate

  The date the service definition is effective to.

- ContractVersionID

  The current version ID of the service definition.

- VersionValidFromDate

  The date this version of the service definition is valid from.

- VersionValidToDate

  The date this version of the service definition is valid to.

Metric

The Metric object contains the information about the metric that hosts the Metric parameter CA Business Service Insight entity. This information includes:

- ClientAPIVersion

  This is the version of the object. This data is useful to know when the object's schema was modified last.

- Name

  This is the metric's name.

- Type

  Always Metric.

- ContractVersionID

  The current version ID of the contract, contract template or service definition in which this metric resides and from which it was launched.

- ParentEntityType

  The type of the metric's hosting entity. Can be: Contract, Service Definition or Contract Template.

UserContextData

The UserContextData object contains the information about the logged-in user. This information includes:

- UserID

  The user ID of the logged-in user.

- UserName

  The user name of the logged-in user.

- Organization

  The organization name of the logged-in user.

- UserNameAtOrganization

  A string that represents the combined value of the user name and organization name.

This information is useful if accessing CA Business Service Insight through its APIs from the launcher page is required.

Customization

The Customization object contains the GetEntity(), Close(), Cancel(), AttachEvent (eventName, eventHandler) and SetWindowTitle (title) methods that are available to the Launch Page. For more information, see JavaScript Methods (see page 324).

JavaScript Events

OnClosing

Fires on the launch page just before the modal dialog is closed and the entity is sent back to CA Business Service Insight for update in GUI. Provides a neat entry to tamper information or validate data. Can also be used to cancel returning data.

JavaScript Methods

The following methods are available on the Customization JavaScript object and are all available to the Launch Page.

GetEntity()

Returns the entity object.

Close()

Fires the ReturningEntity event and closes the modal dialog. Depending on the results of the ReturningEntity event, the entity object might not be sent back to CA Business Service Insight for update.

Cancel()

Closes the modal dialog without updating the entity in CA Business Service Insight.

AttachEvent (eventName, eventHandler)

Wires an event handler to an existing event. Currently relevant only for the “OnClosing” event.

SetWindowTitle (title)

Sets the modal dialog title.

Chapter 10: Examples

This section contains the following topics:

Translation Scripts - User Function Scripts; Examples

Example 1

```
Sub CheckIsUserExists

  If (Tools.IsUserExists("acme@Insight")) Then
    Tools.Log "'acme@Insight' User exist.", "I"
  Else
    Tools.Log "'acme@Insight' User does not exist.", "I"
  End If

  If (Tools.IsUserExists("acme")) Then
    Tools.Log "'acme' User exist.", "I"
  Else
    Tools.Log "'acme' User does not exist.", "I"
  End If

  If (Not Tools.IsUserExists("databaseMtn@INTERNAL")) Then
    Tools.Log "'DataBaseMtn@INTERNAL' User does not exist.", "I"
  Else
    Tools.Log "'DataBaseMtn@INTERNAL' User exist.", "I"
  End If

End Sub

Sub CheckGetUserFullName
  Tools.Log "GetUserFullName", "I"
  Tools.Log Tools.GetUserFullName("aaaa","bbbb"),"I"
  Tools.Log Tools.GetUserFullName("aaaa",""),"I"
  Tools.Log Tools.GetUserFullName("","bbbb"),"I"
  Tools.Log Tools.GetUserFullName("aa@aa","bb@bb"),"I"
End Sub
```

```
Sub CheckGetUserName
  Tools.Log "GetUserName", "I"
  Tools.Log Tools.GetUserName("aaaa@bbbb"),"I"
  Tools.Log Tools.GetUserName("aaaa@bb@bb"),"I"
  Tools.Log Tools.GetUserName("@bbbb"),"I"
  Tools.Log Tools.GetUserName("aa aa b@bbbb"),"I"
  Tools.Log Tools.GetUserName("aaaabbbb"),"I"
End Sub

Sub CheckGetOrganizationName
  Tools.Log "GetOrganizationName", "I"
  Tools.Log Tools.GetOrganizationName("aaaa@bbbb"),"I"
  Tools.Log Tools.GetOrganizationName("aaaa@bb@bb"),"I"
  Tools.Log Tools.GetOrganizationName("@bbbb"),"I"
  Tools.Log Tools.GetOrganizationName("aa aa b@bbbb"),"I"
  Tools.Log Tools.GetOrganizationName("aaaabbbb"),"I"
End Sub

Sub CheckSearchUsers
  SearchUsers "","","",""
  SearchUsers "INACTIVE","","",""
  SearchUsers "ACTIVE","","",""
  SearchUsers "","*i*","",""
  SearchUsers "","*e*","",""
  SearchUsers "","","Test",""
  SearchUsers "","","Bla",""
  SearchUsers "","","","my"
  SearchUsers "","","","*my*"
  SearchUsers "","acm*","","*my*"
End Sub

Sub SearchUsers (status, name, organization,description)
  Dim map

  Set map = Tools.SearchUsers(status, name, organization,description)
  Tools.Log "Search Users:   Name        =" & name
  Tools.log "                Organization=" & organization
  Tools.Log "                status      =" & status
  Tools.Log "                Description =" & description
  Tools.Log map.dump, "I"
End Sub

Sub CheckGetUserDetails
  GetUserDetails "acme@Test"
  'GetUserDetails "XXX@YYY"  ' search for user that does not exist. It must fail.

End Sub
```

```
Sub GetUserDetails (name)
  Dim map
  Set map = Tools.GetUserDetails(name)
  Tools.Log map.dump
End Sub

Sub CheckAddUser
  Dim map
  Set map = Tools.GetUserDetails("acme@Test")

  'Check user already exists
  'Tools.AddUserByMap map

  'Check with duplicate
  map("UserName") = "acme2"
  map("UserPassword") = "acme2"
  map("UserPasswordExpirationInterval") = "50"
  map("UserDescription") = "New description"
  map("UserStatus") = "INACTIVE"

  Tools.AddUserByMap map
  Tools.Commit

End Sub

Sub CheckUpdateUser
  Dim map
  Set map = Tools.GetUserDetails("acme@Test")

  'Check user already exists
  'Tools.AddUserByMap map

  map("UserName") = "acmeKrakover"
  map("UserPassword") = "acme"
  map("UserPasswordExpirationInterval") = "55.5"
  map("UserDescription") = "New description"
  map("UserGroups") = "acme test group"
  map("Roles") = "Supervisors"
  map("ContractParties") = "acme CP Group 123"

  Tools.Log map.dump
  Tools.UpdateUserByMap "acmeKrakover@Test", map
  Tools.Commit

End Sub
```

Example 2 - Running LDAP queries from within a translation script

In order to run LDAP queries from within a translation script, each specific, individual translation script must be configured so that it does **not** run in safe mode. Go to **t_translation_script** using a database access tool and manually change the script.

This can be done using an update query such as the following (change the *where clause* as required):

```
update t_translation_scripts
set safe_mode = 'N'
where script_id = …
```

In order to return the script to run in safe mode as default, reset it by running the following (using the same *where clause* as previously):

```
update t_translation_scripts
set safe_mode = 'Y'
where script_id = …
```

Example 3 - Translation Script - AddException

```
Option Explicit

Sub OnLoad()
  'Tools.log "Translation Script : In OnLoad procedure", "I"
End Sub

Sub OnTranslationEvent(EntryDetails)
  'Tools.log "Translation Script : In OnTranslationEvent function", "I"
End Sub
```

```
Sub Main()
  'Tools.log "Translation Script : In Main procedure", "I"

  Dim map2
  Set map2=Tools.CreateMap

  map2("Exception1")="Exception2"
  map2("ExceptionName")="Exception2"
  map2("ExceptionJustification")="Exception2"
  map2("ExceptionDescription")=""
  map2("ExceptionEffectiveFrom")="01/12/2009"
  map2("ExceptionEffectiveTo")="05/12/2009"
  map2("ExceptionTimeSlot")="Always"
  map2("ContractParties") = Tools.CreateMap
  'Uncomment the following line to assign the exception to contract party:
contractParty1
  'map2("ContractParties")("contractPartyName")="contractParty1"
   map2("Contracts")= Tools.CreateMap
  'Comment out the following line if only assigning the exception to a contract party,
and not a contract. The following key is in the format: ContractPartyNameContractName
  map2("Contracts")("contractParty1contract1")= "contract1"
  map2("Services")=Empty
  map2("DomainCategories")=Empty

Tools.AddException(map2)

Tools.Commit

End Sub

Function Result
  'Tools.log "Translation Script : In Result function", "I"
  'Result =
End Function
```

CMDB Example

```
Option Explicit

' The following constants are names that the script uses for creating new entities.
' In the first running you must:
'     1. Create a New Contract Party called "CMDB Template"
'     2. Verify that a suitable Adapter with a suitable file already run.

' In every run at the same environment you must change the name of
' the constant and the Resource name in the input file of the Adapter.
' CA recommends incrementing the number contained within the
  name, e.g., "CMDB CS3" would become "CMDB CS4"


Const CHANGE_SET_NAME= "CMDB CS"
Const RESOURCE_TYPE1 = "CMDB RT1"
Const RESOURCE_TYPE2 = "CMDB RT2" ' This RT will be used for Attach Function
Const SERVICE = "CMDB Service1"
Const RESOURCE_GROUP1 = "CMDB Group1"
Const RESOURCE_GROUP2 = "CMDB Group2"  ' This RG will be used for Attach Function
Const CONTRACT_PARTY = "CMDB CP1"
Const CONTRACT_PARTY_TEMPLATE = "CMDB Template"
Const RESOURCE_NAME = "CMDB Resource1"
Const SECOND_RESOURCE_NAME = "CMDB Based on CMDB Resource1"

Dim added
Dim translated
Dim ignored
Dim deleted

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''
' The OnLoad sub routine adds several entities to the system, if they
  do not already exist.
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''


Sub OnLoad()
  added = 0
  translated = 0
  ignored = 0
  deleted = 0

  ' Update general details for all the resource groups and resources where their name
starts with YYY
  IntResourcesUpdateGeneralDetails "YYY"

  ' Add two Resource Types
```

```
  IntAddResourceType RESOURCE_TYPE1, "Resource Type was added
  automatically by a script."
  IntAddResourceType RESOURCE_TYPE2, "Resource Type was added
  automatically by a script."

  ' Add Change Set
  IntAddChangeSet CHANGE_SET_NAME, "01/01/2006", "Change Set
  was added automatically by a script."

  ' Add Service
  IntAddService Service, "Service was added automatically by
  a script.", 100

  ' Add Contract Party
  IntAddContractPartyFromTemplate CONTRACT_PARTY, "CP was
  added automatically by a script.", "Provider", "01/01/2006"


  'Add resource groups
  IntAddResourceGroup RESOURCE_GROUP1, CHANGE_SET_NAME, _
                 "Resource Group was added automatically by
                  a script.", SERVICE, CONTRACT_PARTY

  IntDuplicateResourceGroup RESOURCE_GROUP1, RESOURCE_GROUP2,
                          CHANGE_SET_NAME

  'Add resources
  IntAddResource RESOURCE_NAME, CHANGE_SET_NAME, _
            "Resource was added automatically by a script.", _
            RESOURCE_TYPE1,RESOURCE_GROUP1, SERVICE, CONTRACT_PARTY

  IntDuplicateResource RESOURCE_NAME, SECOND_RESOURCE_NAME, CHANGE_SET_NAME

  Tools.Commit

End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''
' Get one translation entry. The details are in EntryDetails parameter and the relevant
fields are:
' - EntryId - the translation entry ID. The Tools.TranslateEntry, Tools.IgnoreEntry
and Tools.DeleteEntry
'   methods get this id.
' - FieldValue(1) - the value to be translated.
' The method gets the value from FieldValue(1) and its process depends on the field
value:
' - If there is a resource with the same name then translate to this resource
' - When it starts with "Ignore me" - ignore the entry
```

```
' - When it starts with "Delete me" - delete the entry
' - When it starts with A then add a resource with all details.
' - When it starts with B then add a resource with only resource
'   type and attach the other data with the specific methods.
' - When it starts with F'F' then add a resource with only resource type and then attach
two resource \
'   types and two resource groups.
' - In all other cases add the resource and translate the entry
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''


Sub OnTranslationEvent(EntryDetails)
  Tools.log "Translation entry details : " & EntryDetails.dump, "D"

  Dim TranslateFrom
  Dim ResourceName
  Dim EntryID

  Dim ResourceTypesMap
  Dim ResourceGroupsMap
  Dim ResourceMap

  Set ResourceTypesMap = Tools.CreateMap
  Set ResourceGroupsMap = Tools.CreateMap
  Set ResourceMap= Tools.CreateMap

  ResourceTypesMap(RESOURCE_TYPE1) = RESOURCE_TYPE1
  ResourceTypesMap(RESOURCE_TYPE2) = RESOURCE_TYPE2

  ResourceGroupsMap(RESOURCE_GROUP1) = RESOURCE_GROUP1
  ResourceGroupsMap(RESOURCE_GROUP2) = RESOURCE_GROUP2

  TranslateFrom = EntryDetails.FieldValue(1)
  ResourceName= "CMDB " & TranslateFrom
  Tools.log "Resource name : " & ResourceName, "D"
  EntryID= EntryDetails.EntryId

  ' If the resource already exists then translate the entry to the resource
  If Tools.IsResourceExists (ResourceName) Then
    Tools.TranslateEntry EntryID,ResourceName
    translated = translated + 1
    Tools.Log "The entry id " & EntryID & " Translated to the resource: " &
ResourceName,"D"

  ' If the resource name starts with 'Ignore me' substring then ignore the entry
  ElseIf mid(TranslateFrom ,1,9) = "Ignore me" Then
    Tools.IgnoreEntry EntryID
```

```
    ignored = ignored + 1
    Tools.Log "The entry id " & EntryID & " (resource: " & ResourceName & ")
ignored","D"

  ' If the resource name starts with 'Delete me' substring then ignore the entry
  ElseIf mid(TranslateFrom ,1,9) = "Delete me" Then
    Tools.DeleteEntry EntryID
    deleted = deleted + 1
    Tools.Log "The entry id " & EntryID & " (resource: " & ResourceName & ")
deleted","D"

  ' If the resource name starts with 'A' then add resource with all details
  ElseIf mid(TranslateFrom ,1,1) = "A" Then
    IntAddResource ResourceName, CHANGE_SET_NAME, "Resource was added automatically
by a script", _
                RESOURCE_TYPE1,RESOURCE_GROUP1, SERVICE, CONTRACT_PARTY
    added = added + 1
    Tools.TranslateEntry EntryID,ResourceName
    translated = translated + 1

  ' If the resource name starts with 'B' then add resource only with resource type
and attach the
  ' other data with the specific methods.
  ElseIf mid(TranslateFrom ,1,1) = "B" Then
    IntAddResource ResourceName, CHANGE_SET_NAME, "Resource was added automatically
by a script", _
                RESOURCE_TYPE1, Null, Null, Null
    added = added + 1
    Tools.TranslateEntry EntryID,ResourceName
    translated = translated + 1

    Tools.AttachResourcesToContractParties ResourceName, CHANGE_SET_NAME,
CONTRACT_PARTY
    Tools.AttachResourcesToResourceGroups  ResourceName, CHANGE_SET_NAME,
RESOURCE_GROUP1
    Tools.AttachResourcesToServices        ResourceName, CHANGE_SET_NAME, SERVICE
    IntLogEntityDetails "resource", "new", Tools.GetResourceDetails
(ResourceName,CHANGE_SET_NAME).Dump

  ' If the resource name starts with 'F' then add resource only with resource type
and then attach
  ' two resource types, two resource groups.
  ElseIf mid(TranslateFrom ,1,1) = "F" Then
    IntAddResource ResourceName, CHANGE_SET_NAME, "Resource was added automatically
by a script", _
                RESOURCE_TYPE1, Null, Null, Null
    added = added + 1
    Tools.TranslateEntry EntryID,ResourceName
    translated = translated + 1
```

```
    Tools.AttachResourcesToResourceTypes   ResourceName, CHANGE_SET_NAME,
ResourceTypesMap
    Tools.AttachResourcesToResourceGroups   ResourceName, CHANGE_SET_NAME,
ResourceGroupsMap

  ' In all the other cases add the resource and translate the entry
  Else
    ResourceMap("ResourceName") = ResourceName
    ResourceMap("ResourceDisplayName") = ResourceName
    ResourceMap("ChangeSetName") = CHANGE_SET_NAME
    ResourceMap("ResourceDescription") = "Resource was added automatically
(AddResourceByMap) by a script"
    ResourceMap("ResourceTypes") = ResourceTypesMap
    ResourceMap("ResourceGroups") = ResourceGroupsMap
    ResourceMap("Services") = SERVICE
    ResourceMap("ContractParties") = CONTRACT_PARTY
    ResourceMap("Effective") = "Yes"
    ResourceMap("CustomAttributes") = Null


    Tools.AddResourceByMap ResourceMap
    added = added + 1
    Tools.TranslateEntry EntryID,ResourceName
    translated = translated + 1

    IntLogEntityDetails "resource", "new", Tools.GetResourceDetails
(ResourceName,CHANGE_SET_NAME).Dump
  End If

  Tools.commit

End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''

Sub Main()
  'Tools.log "Translation Script : In Main procedure", "I"
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''

' Add change set. The function checks if the change set already exists.
Sub IntAddChangeSet(Name, EffectiveDate, Description)
  If Not Tools.IsChangeSetExists(Name) Then
    Tools.AddResourceVersion Name, EffectiveDate,Description
    Tools.Log "Change Set was added successfully: " & Name, "D"
```

```
    IntLogEntityDetails "change set", "new", Tools.GetChangeSetDetails(Name).Dump
  End If

End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''

' Add resource type. The function checks if the resource type already exists.
Sub IntAddResourceType(Name, Description)
  If Not Tools.IsResourceTypeExists (Name) Then
    Tools.AddResourceType Name, Description, Null
    Tools.Log "Resource Type was added successfully: " & Name, "D"
    IntLogEntityDetails "resource type", "new",
Tools.GetResourceTypeDetails(Name).Dump
  End If
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''

' Add Service. The function checks if the service already exists.
Sub IntAddService(Name, Description, Fee)
  If Not Tools.IsServiceExists (Name) Then
    Tools.AddService Name, Description, Fee, Null
    Tools.Log "Service was added successfully: " & Name, "D"
    IntLogEntityDetails "service", "new", Tools.GetServiceDetails(Name).Dump
  End If
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''

' Add Contract Party. The function checks if the contract party already exists.
Sub IntAddContractPartyFromTemplate(Name, Description, Provider, RegistrationDate)

  Dim ContractPartyMap

  If Not Tools.IsContractPartyExists (Name) Then
    Set ContractPartyMap = Tools.GetContractPartyDetails (CONTRACT_PARTY_TEMPLATE)
    ContractPartyMap("ContractPartyName")=Name
    ContractPartyMap("ContractPartyDescription")=Description
    ContractPartyMap("ContractPartyType")=Provider
    ContractPartyMap("ContractPartyRegistrationDate")=RegistrationDate
    Tools.AddContractPartyByMap ContractPartyMap
    Tools.Log "Contract Party was added successfully: " & Name, "D"
    IntLogEntityDetails "contract party", "new",
Tools.GetContractPartyDetails(Name).Dump
  End If
```

```
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''

' Add Resource Group. The function checks if the resource group already exists.
Sub IntAddResourceGroup(Name, ChangeSetName, Description, Service, ContractParty)
  If Not Tools.IsResourceGroupExists (Name) Then
    Tools.AddResourceGroup Name, ChangeSetName, Description, Null, Service,
ContractParty
    Tools.Log "Resource Group was added successfully: " & Name, "D"
    IntLogEntityDetails "resource group", "new", Tools.GetResourceGroupDetails
(Name,ChangeSetName).Dump
  End If
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''

' Duplicate Resource Group. The function checks if the resource group already exists.
Sub IntDuplicateResourceGroup(OldName, NewName, ChangeSetName)

  Dim ResourceGroupMap

  If Not Tools.IsResourceGroupExists (OldName) Then
    Tools.Log "Resource Group " & OldName & " does not exist. ", "E"

  Else
    If Not Tools.IsResourceGroupExists (NewName) Then
      Set ResourceGroupMap = Tools.GetResourceGroupDetails (OldName,ChangeSetName)
      ResourceGroupMap("ResourceGroupName")=NewName
      Tools.AddResourceGroupByMap ResourceGroupMap
      Tools.Log "Resource Group was duplicated successfully: " & NewName, "D"
      IntLogEntityDetails "resource group", "new", Tools.GetResourceGroupDetails
(NewName,ChangeSetName).Dump
    End If
  End If
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''

' Add Resource. The function checks if the resource already exists.
Sub IntAddResource(Name, ChangeSetName, Description, ResourceType, ResourceGroup,
Service, ContractParty)
  If Not Tools.IsResourceGroupExists (Name) Then
    Tools.AddResource Name, ChangeSetName, Description, ResourceType,
ResourceGroup, Service, ContractParty
    Tools.Log "Resource was added successfully: " & Name, "D"
```

```
    IntLogEntityDetails "resource", "new", Tools.GetResourceDetails
(Name,ChangeSetName).Dump
  End If
End Sub

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''

' Duplicate Resource. The function checks if the resource group already exists.
Sub IntDuplicateResource(OldName, NewName, ChangeSetName)

  Dim ResourceMap

  If Not Tools.IsResourceExists (OldName) Then
    Tools.Log "Resource " & OldName & " does not exist. ", "E"

  Else
    If Not Tools.IsResourceExists (NewName) Then
      Set ResourceMap = Tools.GetResourceDetails (OldName,ChangeSetName)
      ResourceMap("ResourceName")=NewName
      Tools.AddResourceByMap ResourceMap
      Tools.Log "Resource was duplicated successfully: " & NewName, "D"
      IntLogEntityDetails "resource", "new", Tools.GetResourceDetails
(NewName,ChangeSetName).Dump
    End If
  End If
End Sub


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''
' Run on all the resource groups and resources with the given prefix and
' get their general details, change the details, and update them in the DB
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''

Sub IntResourcesUpdateGeneralDetails(NamePrefix)

  Dim map
  Dim generalDetails
  Dim resourceName
  Set map = Tools.SearchResourceGroups ("", NamePrefix + "*")
  If (map.count >0) Then
    For Each resourceName In map
      Set generalDetails = Tools.GetResourceGroupGeneralDetails (resourceName)
      Tools.Log "General details before update:"
      Tools.Log GeneralDetails.dump

      GeneralDetails("ResourceGroupName") = resourceName + " Renamed"
```

```
      GeneralDetails("ResourceGroupDisplayName") = resourceName + " Renamed"
      GeneralDetails("ResourceGroupDescription") = "New description"

      Tools.UpdateResourceGroupGeneralDetails resourceName, GeneralDetails
      Tools.Log "General details after update"
      Tools.Log GeneralDetails.dump
    Next
  End If

  Set map = Tools.SearchResources("", NamePrefix + "*")
  If (map.count >0) Then
    For Each resourceName In map
      Set generalDetails = Tools.GetResourceGeneralDetails(resourceName)
      Tools.Log "General details before update:"
      Tools.Log GeneralDetails.dump

      GeneralDetails("ResourceName") = resourceName + " Renamed"
      GeneralDetails("ResourceDisplayName") = resourceName + " Renamed"
      GeneralDetails("ResourceDescription") = "New description"

      Tools.UpdateResourceGeneralDetails resourceName, GeneralDetails
      Tools.Log "General details after update"
      Tools.Log GeneralDetails.dump
    Next
  End If

End Sub

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''

Sub IntLogEntityDetails(EntityType, ActionType, Details)
  Tools.Log "The " & ActionType & " " & EntityType & " details: " & chr(10) & chr(13)
& Details,"D"
End Sub

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'''''''''''''''

Function Result
  Result = added & " resources had been added, " & _
           translated & " resources had been translated, " & _
           ignored & " resources had been Ignored and " & _
           deleted & " resources had been deleted."
End Function
```

Launcher Page Examples

Example 1

The following code example uses the launcher page objects and exhibits updating the entity, data validation and also canceling the operation.

```
var customizationHelper = new window.top.Customization();

function onLoad()
{
    // update launch page GUI
    var entity = customizationHelper.GetEntity();
    if ((entity.EntityType == "CustomAttribute") &&
        (entity.Name == "Location") &&
        (entity.ValueType == "Text"))
    {
        var textbox = document.getElementById("txtValue");
        textbox.value = entity.Value;

       // can go to server here in order to populate
       // the page with data
    }
}

function Close()
{
    var entity = customizationHelper.GetEntity();
    var textbox = document.getElementById("txtValue");

    // data validation
    if ((entity.Name == "Location") &&
        (textbox.Value.indexOf("Home") != -1))
    {
        alert ("Home is not allowed as location!");
    }
    else
    {
        // update entity before sending back to OG
        entity.Value = textbox.value;

        customizationHelper.Close();
    }
}

function Cancel()
{
     // close the launch page and don't update entity
     customizationHelper.Cancel();
}
```

Example 2

The following code example uses the launcher page objects.

```
var customizationHelper = new window.top.Customization();
function exploreEntityContextData()
     {
        var entity = customizationHelper.GetEntity();

        if(entity.contextData)
        {
             var contextData = entity.contextData;

             if(contextData.Contract)
             {
                var contract = contextData.Contract;

                var info = "";

                info += writeInfoItem("Type", contract.Type);
                info += writeInfoItem("ContractName", contract.Name);
                info += writeInfoItem("ContractType", contract.ContractType);
                info += writeInfoItem("ContractTypeName",
contract.ContractTypeName);
                info += writeInfoItem("ContractVersionID",
contract.ContractVersionID);
                info += writeInfoItem("ContractPartyReceiverName",
contract.ContractPartyReceiverName);
                info += writeInfoItem("ContractPartyProviderName",
contract.ContractPartyProviderName);

                contractInfo.innerHTML = info;
             }
        }
     }

     function writeInfoItem(fieldName, value)
     {
        return "<b>" + fieldName + ":</b> " + value + "<br>";
     }
```

Glossary

Adapter

The interface between CA Business Service Insight and third-party data sources. Adapters translate data taken from these data sources into a format that can be used for service level calculations that are provided to contract parties.

Agent

An object that represents a metric in a given time unit.

Alert

Notifications to users about events taking place in the system. Alerts enable users to take corrective action to avoid deviating from contracts, incurring penalties and so on.

Alert Device

Devices in the network that receive alert messages via e-mail, popups or SMS or via a third-party program.

Alert Profile

A set of parameters that defines the conditions under which an alert message is triggered, the users to whom it is issued and how it is sent.

Archived Contract

A contract whose data has been moved to the archive. Archived contracts are not included in calculations and cannot be viewed in reports.

Audit Trail

A chronological record of CA Business Service Insight user and system activities, which are saved in the system. The audit can later be reviewed to identify users actions on the system or processes which occurred on the system.

Booklet

RTF-based file that may include contract data and related reports in a convenient format using pre-defined and configurable design templates

Business Logic Modules

Library of script functions that can be used in Business Logic.

Business Logic Template

Predefined Business Logic formulas that can be used to define new Business Logic formulas.

Change Set

A set of changes to the service provider's resources map.

Clustered Metric

A type of metric that applies to multiple resources or resource groups.

Compound Report

Multiple regular reports that are displayed as multiple series on a single chart

Consumption

A metric type that calculates consumption. Used mainly for the financial module. By using this metric type, it is possible to see in the reports the consumption and the prices separately. The consumption can be calculated in the Price Item metric as well.

Consumption Metric

A metrics that provides the ability to view consumption and prices separately in a report.

Contract Audit Trail

A log of all of the activities of a contract.

Contract Author

The user who is responsible for creating a certain contract.

Contract Party

The customer or provider with whom the contract is signed. A contract party can also represent an internal entity within a larger organization.

Contract Party Group

Logically defined group of contract parties (customers). Referring to a group instead of an individual contract party streamlines contract creation. A contract party can belong to more than one contract party group.

Contract Template

Predefined contract used to create a new contract.

Contract-level Parameter

A parameter of a regular contract, contract template and service level template that is being used by metrics within the business logic.

Current Status Engine

A standalone process that calculates current status metrics. There is no limit to the number of instances allowed to run on one or more machines.

Dashboard

A user interface intended for high-level managers that organizes and presents real-time information and reports in an easy-to-read manner.

Dashboard Map

The layout of the dashboard. The arrangement of widgets in the dashboard.

Data Event

Events generated by CA Business Service Insight adapters

Derived Metric

A metric that was created from a contract template or service level template.

Domain Category

A specific aspect of the service level agreed upon in the contract. For example, you might have a service domain called Help Desk and a domain category such as Response Time of the Help Desk, which describes that particular aspect of Help Desk service. The service provider's system administrator usually defines the domain categories. CA Business Service Insight also supplies predefined domain categories.

Event Annotations

Additional information about a specific event. Event Annotations are set automatically or manually (within the reports).

Event Types

An event is a measurement done on a resource by a third-party measuring tool and then translated by the adapter to a format usable by CA Business Service Insight. Event types determine how these events are defined and formatted for CA Business Service Insight.

Exception

Period of time that does not count towards the calculation of service levels. For example, downtime.

Free-Form Report

A report that is created by a user defined SQL-statement based on CA Business Service Insight database or other external databases.

Granularity

Granularity determines the additional time units the metric author wants to get result for excluding the tracking period of the metric.

Group Report

A report that places several reports side-by-side on one page.

Informational Metric

A metric that performs informational calculation to be used for reporting purposes only.

Interim Metric

A metric that can generate events for the purpose of calculating events. Interim metrics cannot be calculated.

KPI

Key Performance Indicator. A significant measure used on its own, or in combination with other key performance indicators, to monitor how well a business or a service is achieving its quantifiable objectives.

KQI

Key Quality Indicator. A significant measure used on its own, or in combination with other key performance indicators, to monitor how well a business or a service is achieving its quality objectives.

Manual Root-Cause Comment

A comment, set manually by the report viewer, to explain or add additional information to a service level result of specific metric. Manual root-cause comments can be shared by all report viewers of the same metric.

Metric

A combination of several parameters defining the target service level for a certain service at a certain time. Each metric is attached to a single service domain. Metric fields and attributes include domain category, service level formula, service, timeslot, service level target and tracking period.

Metric Event

An event generated by a metric in CA Business Service Insight.

Metric Type

Describes the reason for calculation of a certain metric and defines the list of fields and their behavior in terms of mandatory fields and defaults that a metric of a certain type should have.

Metric Wizard

A user-friendly interface developed to allow users to easily modify metrics the first time the metric is added to the contract from a service level template.

Metric-level Parameters

Parameters of a metric.

Objective Statement

The logical description of a metric containing the parameters that define the metric. Objective statements are displayed in the CA Business Service Insight GUI to provide effective capturing of the metric essence.

OLA

Operational Level Agreement. An OLA is an SLA in which the supplier is the internal organization and the customers are internal business units.

Package

A collection of service level templates and templates packed together to be installed and unpacked in another CA Business Service Insight instance.

Parameter

A value that can be set outside of the business logic to affect the business logic formula. Parameters are being used by the business logic to determine the service level result. A parameter can be of type text, list, number, date or table. Examples of parameters are thresholds and resource names.

Penalty

The compensation owed to the contract party if the service level target is not met within a defined tracking period. Penalties can be turned on or off in CA Business Service Insight.

Price Item

A metric that calculates the prices. The prices can be based on the consumption or defined as fixed prices.

Ratio

A type of Unit.

Raw Data Event

An event generated by raw data in CA Business Service Insight.

Received Events

A list of events received from other metrics shown in the Business Logic Scope window when clicking on the Received Events tab.

Regional Options

Specific details for the locality of the contract party. Parameters include language, currency, time difference from GMT, daylight saving time, date format and currency.

Related Metric

A metric that is referenced as a target in a relation.

Relation

An entity describing a connection of a certain type established between two metrics and having some properties.

Report Wizard

Graphical User Interface (GUI) used to define report parameters.

Report-Group Report

A report that encapsulates multiple regular reports, placed side-by-side

Reports Folder

Reports Folder is a container used to group together reports related to each other in some way or another.

Resource

A single element of the service provider's total infrastructure. Examples of resources include individual servers, switches, hubs, routers, a Help Desk or any other measurable element. Multiple resources can be defined as part of a resource group, which is then treated as another resource itself.

Resource Allocation

Allocating a resource to a service direct the resource events stream to this service. A resource can be allocated to a service, contract party, type or group.

Resource Effective Date

The date from which CA Business Service Insight may use the resource. The resource effective date is defined by the resource version that includes the resource. The resource will not be recognized by any resource version that has an effective date prior to the resource version in which the resource was created.

Resource Entry Date

The date the resource was entered into CA Business Service Insight. This should not be confused with the Resource Effective Date.

Resource Group

A collection of resources that are grouped together as a logical unit. A resource group can contain one or more individual resource or resource group.

Resource Type

A built-in category of resources. Resource must be allocated to at least one resource type.

Role

A set of responsibilities, activities and authorizations. The user role defines the view mode of CA Business Service Insight Every CA Business Service Insight user is assigned one or more roles which determine which actions the user can perform within CA Business Service Insight. Actions that cannot be performed by the role are removed from the CA Business Service Insight user interface when that user accesses the application.

Root-Cause Comment

A comment set within the business logic to explain the service level results. The root-cause comments are associated with metrics.

Service Catalog

A collection of all the information about services, domains and units, templates library and service level templates in CA Business Service Insight.

Service Level Template

A service definition is a predefined set of services and metrics, used to facilitate the creation and modification of contracts according to common business processes.

Simple Report

A means to analyze results calculated by CA Business Service Insight based on a rich set of criteria.

SLO

Service Level Objective. Used to measure contractual agreements.

Tracking Period

The time interval during which CA Business Service Insight measures the provided service level versus a service level objective defined in the contract. The measurement taken during this period determines whether there is a deviation from the defined objective (as defined by the business logic) and whether any penalties or bonuses have been incurred. The tracking period also determines how business reports are generated.

Translation Entry

Represents a definition of source and destination values that are used by the translation table.

Translation Script

A script that can be used to ease the maintenance of translations and prevent errors.

Translation Table

A mechanism used for translating values from the data that comes from the raw data to the data defined in the system.

Translations

The conversion of data collected by adapters from their data sources into entities that have been defined in CA Business Service Insight.

Underpinning Contract

Underpinning contract. A contract with an external supplier covering delivery of services that support the organization in their delivery of services.

Unit

A catalog of measurable units. Examples include percentage and currency.

Unit of Measurement

A unit of measurement for all the metrics defined for a domain category.