XTM International

XTM Connect SDK

Documentation

Better Translation Technology

Documentation for XTM Connect SDK

Published by XTM International Ltd.

Updated November 2018

TABLE OF CONTENTS

1 Introduction

1.1 Overview of XTM Connect

The XTM Connect SDK is available as both SOAP and REST APIs. The Soap API provides more complete functionality, but is more complex to implement, while the REST API has a limited functionality but is quicker and easier to implement. You can find more information about the REST API at https://www.xtm-cloud.com/rest-api/.

If you wish to test the XTM Connect SDK please contact sales@xtm-intl.com who can set up a sandbox environment and provide you with the requisite URLs.

The web services are generally designed to be passive. They do not need know anything about the customer service and will only call the customer service after specified workflow events. The web services mainly respond to calls from the customer.

For file transfers, the web services can accept a URL, encoded BASE64 binary information or an MTOM file attachment.

There are two key concepts for the web services:

- Project - this is the overall project that is created.
- Job - a project may comprise one or more language pairs (source language and target language). Each language pair is called a 'job' within a project.

The following steps describe how to use the XTM Connect SDK:

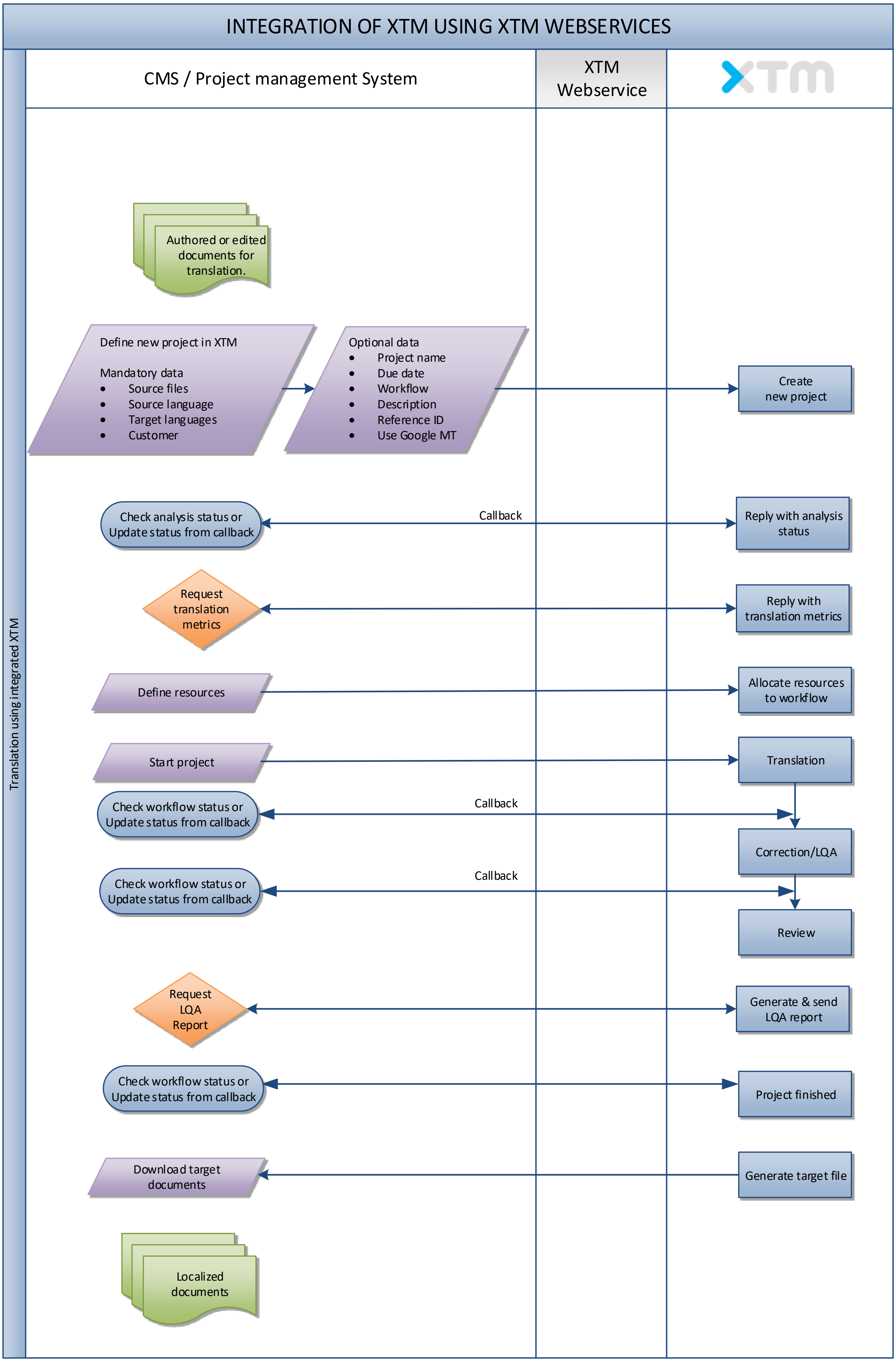
1. Create a project. Certain data is mandatory to create a project and other data is optional.

Mandatory data	Optional data
• Source files	• Project name
• Source language	• Due date
• Target languages	• Workflow
• Customer	• Description
	• Reference ID
	• Use Google MT

2. Check if the project/or particular job has finished its analysis stage
3. Get the job/project metrics if required. Depending on the metrics the Customer may decide not to proceed with the project.
4. Assign linguists.
5. Start or delete the Project.
6. Check the status of the Project.
7. Generate the target file, XLIFF or QA report and download it (you can either request URL or MTOM download).

Steps 2. to 5. are optional since the project manager can perform these tasks from within XTM.

The diagram below shows a typical integration.


INTEGRATION OF XTM USING XTM WEBSERVICES
Translation using integrated XTM
CMS / Project management System
XTM Webservice
XTM
Authored or edited documents for translation.
Define new project in XTM
Mandatory data
Source files
Source language
Target languages
Customer
Optional data
Project name
Due date
Workflow
Description
Reference ID
Use Google MT
Create new project
Check analysis status or Update status from callback
Callback
Reply with analysis status
Request translation metrics
Reply with translation metrics
Define resources
Allocate resources to workflow
Start project
Translation
Check workflow status or Update status from callback
Callback
Correction/LQA
Check workflow status or Update status from callback
Callback
Review
Request LQA Report
Generate & send LQA report
Check workflow status or Update status from callback
Project finished
Download target documents
Generate target file
Localized documents

Integration of XTM using XTM Web services

1.2 The Benefits of Integrating XTM Using XTM Connect SDK

The XTM Connect SDK provides comprehensive web services to allow the easy integration of XTM with existing CMS or project management systems.

- The XTM Connect SDK offers complete flexibility to suit the way you wish to work. For example:
    - XTM can be treated as a translation “black box” that handles all the issues related to translation such as resource allocation and workflow.
    - The workflow, translation resources and other parameters can be defined in the third party system and then sent to XTM with the documents for translation. Status updates and reports including project metrics can then be obtained from XTM.
- The XTM Connect SDK reduces the time and effort required for the integration to a minimum.
- XTM can be integrated with any system as long as it supports web services. It is not important what programming language has been used to write the third party system.
- Updates to XTM will be compatible with the XTM Connect SDK. This means you will have instant access to the latest features in XTM as soon as they are developed by XTM International.
- The integration of XTM into your project or translation workflow streamlines processes, speeds throughput times and assists project management.
- The XTM Connect SDK is available for XTM Suite and XTM Cloud.

2 API Details

2.1 Definitions

1. Definitions of common objects

- LoginAPI: userId, username, password, client, integrationKey
  Required for all web service calls. Please set integrationKey if provided by XTM.

- XTMCustomerDescriptorAPI: id, name, externalId
  The Customer descriptor is used to specify a customer. Firstly the customer ID is checked, then the customer name, finally the customer external ID.

- XTMDomainDescriptorAPI: domain, domainName
  The Domain descriptor is used to specify a domain. Firstly domain definition from XTM_DOMAINS enumeration type is checked, then the custom domain name.

- XTMFileDescriptorAPI: id
  The File descriptor is used to specify a file.

- XTMJobDescriptorAPI: id, externalId (deprecated, please use integrationId), integrationId
  The Job descriptor is used to specify a job. Firstly the job ID is checked, then the job integration ID.

- XTMProjectDescriptorAPI: id, externalId (deprecated, please use integrationId), integrationId
  The Project descriptor is used to specify a project. Firstly the project ID is checked, then the project integration ID.

- XTMUserBaseDescriptorAPI: id, name
  The Base user descriptor is used to specify a user. Firstly the user ID is checked, then the user name.

- XTMUserDescriptorAPI: id, name, actorType
  User descriptor is used to specify a linguist or LSP. Firstly the user ID is checked, then the user name. Actor type allows to specify linguist or LSP.

- XTMWorkflowDescriptorAPI: id, workflow, workflowName
  Workflow descriptor is used to specify a workflow. Firstly the workflow ID is checked, then the workflow definition from XTM_WORKFLOWS enumeration type, finally the custom workflow name.

- XTMWorkflowStepDescriptorAPI: id, orderNumber, workflowStep, workflowStepName
  Workflow step descriptor is used to specify a workflow step. Firstly the workflow step ID and orderNumber are checked and then the workflow step definition from XTM_WORKFLOW_STEP enumeration type and the custom workflow step name.
  To view the workflow step ID go to Configuration tab -> Settings -> Workflow -> Workflow steps and mouse over the information icon.

- XTMTemplateDescriptorAPI: id, externalId
  The Template descriptor is used to specify a project template. Firstly the template ID is checked, then the template external ID.

- XTMDeleteFileDescriptorAPI: fileName
  This descriptor is used to specify a file to delete.

- XTMTMBaseFileDescriptorAPI: id
  This descriptor is used to specify TM import or TM export file.

- XTMPenaltyProfileDescriptorAPI: id, name
  This descriptor is used to specify a TM or Term penalty profile. Firstly the profile ID is checked, then the profile name.

- XTMTagDescriptorAPI: id, name
  This descriptor is used to specify a tag group and the list of tags. Firstly the tag group ID is checked, then the tag group name and then the tags which are defined by XTMTagValueDescriptorAPI object.

- XTMTagValueDescriptorAPI: id, name
  This descriptor is used to specify a tag. Firstly the tag ID is checked, then the tag name.

- XTMCostDescriptorAPI: costBaseDescriptor, projectDescriptor, source
  This descriptor is used to specify a cost. Firstly the ID from the costBaseDescriptor is checked, then the project descriptor with a cost calculation source, finally the project descriptor.

- XTMCustomFieldDescriptorAPI: id
  This descriptor is used to specify a custom field.

- XTMCustomFieldValueAPI: ids, value, dateValue, booleanValue
  This object is used to specify a custom field value. Custom fields can be set to have multiple values
  - The ids in a multiple selection or a dropdown custom field are used to choose the options
  - The booleanValue field is used for checkbox custom fields,
  - The dateValue for custom fields with dates
  - The value field is used for other types of custom fields.

- XTMCustomFieldAPI: customFieldDescriptor, customFieldValue
  This descriptor is used to specify a custom field and its value.

- XTMAnalysisTemplateDescriptor: id
  This descriptor is used to specify an analysis template.

2. Enumeration types

- XTM_WORKFLOW_STEP - list of possible workflow steps

```
/** define for ‘translate 1’ step. */
TRANSLATE1,

/** define for ‘review 1’ step. */
REVIEW1,

/** define for ‘review 2’ step. */
REVIEW2,

/** define for ‘correct 1’ step. */
CORRECT1,

/** define for ‘correct 2’ step.*/
CORRECT2;
```

- ActorType - list of possible linguist types

  ```
  /** define for linguist type. Default value if not set. */
  INTERNALLINGUIST,

  /** define for LSP type. */
  LSP
  ```

- LANGUAGE_CODE
  Contains languages in the IANA format where a single language subtag composed of two letters is followed by a region subtag composed of two letters, for example: en_GB, pl_PL, ru_RU.

2.2 Roles and Web Service access

The LoginAPI object is required for all calls.

Access to the various methods in the XTM Web Services is controlled by the logon credentials. There are three roles that reflect the functionality in the GUI of the application

ROLE	AVAILABLE WEB SERVICE FUNCTIONALITY	WEBSERVICE
Administrator	Create users Obtain the URL to translation properties of system and customer	Project Manager web service
Internal project manager	Create users, customers and projects Create customer project managers Select which TM to use while creating project Obtain project metrics Manage the workflow Check the status of the project Obtain additional information about the project Download files Upload an XLIFF file Obtain the URL to XTM Terminology manager Obtain the URL to XTM TM manager Obtain the URL to XTM Editor Obtain the URL of XTM Editor for a specific XLIFF file Obtain translator statistics Obtain the URL to Project Editor Obtain base information about XTM Obtain list of supported file types Set project activity Obtain the URL to translation properties of customer	Project Manager web service
Customer project manager	Create projects Obtain project metrics Manage the workflow Check the status of the project Obtain additional information about the project Download files Obtain translator statistics Obtain the URL to Project Editor Obtain base information about XTM Obtain list of supported file types Set project activity	Customer web service

2.3 File transfer

For file transfers, the XTM web service can accept a URL, encoded BASE64 binary information or an MTOM file attachment.

URL – address of a file which can be downloaded by XTM or a client
BASE64 –encoded file in BASE64 format which is included in a web service message
MTOM – file sent as an attachment

To achieve this functionality XTM web services work with two protocols:

1. XML
    - Example: /projectmanager/XTMWebService
    - MTOM disabled
    - Possible file transfers:
        - URL - methods ended with suffix ”URL”
        - BASE64 - methods ended with suffix ”MTOM”

When the MTOM option is disabled for web services, then all the methods designed for MTOM, work in a standard non-optimized way – hence methods still have the suffix “MTOM”

2. XOP (XML – binary Optimized Packaging)
    - Example: /projectmanager/xop/XTMWebService
    - MTOM enabled
    - Possible file transfers:
        - URL - methods ended with suffix ”URL”
        - MTOM - methods ended with suffix ”MTOM”

Please choose MTOM to send or download a large number of files from XTM. Then the size of such documents sent via web services will be much less than using BASE64 and no additional encoding is required. Please choose BASE64 when your integration programming language does not support MTOM (for example PHP).

3 API methods - SOAP

3.1 Customers

1. createCustomer()

   A customer can be created by specifying the XTMCustomerAPI object. The information is divided into the following groups:

   - customer name, external descriptor and VAT number
   - address details
   - instant messaging identifiers
   - additional settings
   - custom fields

   Most of the fields are not mandatory however the following must be specified:
   - name

   XTMExternalCustomerDescriptorAPI: externalId

   External customer descriptor needs to be specified to use customer external ID in other methods instead of customer ID from XTM.

2. updateCustomer()

   This method can be used to update the following customer data:
   - name
   - address
   - custom fields

3. updateCustomerActivity ()

   This method can be used to delete the specified customers. It requires a list of customer descriptors and activity (XTM_CUSTOMER_ACTIVITY).

   XTM_CUSTOMER_ACTIVITY - list of possible activities

   - DELETE
   Deletes customer

4. findCustomer()

   This method can be used to search for customers. It is necessary to specify search parameters in the XTMFindCustomerAPI object. This method returns all customers with details matching the specified criteria. Possible criteria are: list of XTMCustomerDescriptorAPI objects, XTMCustomerPMDescriptorAPI and XTMCustomerActivityFilterEnum.

   XTMCustomerActivityFilterEnum - the list of customer activity statuses available during the search for customers

   - ONLY_ACTIVE - Search only for active customers
   - ALL - Search for active and deactivated customers

5. copyCustomer()

   This method can be used to copy any or all of the following data between customers:

   - project manager
   - estimates factors
   - language combinations
   - domains

   Each field to copy is specified by the XTMCustomerCopyFieldType object.

   XTMCustomerCopyFieldType – the list of possible copy types

```
/** Field will not be copied (default value if not set). */
NONE,

/** Field will be copied. */
FULL
```

3.2 Users

1. createUser()

A user can be created by specifying the XTMUserAdditionAPI object. The information is divided into the following groups:

- user details (contains also: address details and instant messaging identifiers)
- roles
- language combinations
- qualifications
- domains
- terminology rights
- roles descriptors (can be used instead of roles)
- custom fields

Most of the fields are not mandatory however the following must be specified:

- user details
  - username
  - password
  - first name
  - last name
  - e-mail address
- roles

Language Combinations are specified by specifying the Source language and Target language using language codes.

USER_ADDITION_ROLE_API – list of possible roles

- TRANSLATOR
  Language Combinations group is mandatory.
- REVIEWER
  Language Combinations group is mandatory.
- CORRECTOR
  Language Combinations group is mandatory.
- TERMINOLOGY_EXPERT
  Terminology rights group is mandatory. If terminology expert is not a global expert then term customer descriptor is also required.
- PROJECT_MANAGER
  There is an option to limit projects visibility only to projects created by this user (XTMUserProjectsVisibilityEnum).
- TM_EXPERT

XTMUserProjectsVisibilityEnum - possible options for projects visibility for Project Manager

```
/** Only projects created by this user. */
CREATOR,
```

2. updateUser

This method can be used to update the following user data:

- user details: first name, last name, username, password, address
- custom fields

3. updateUserActivity()

This method can be used to delete the specified users or LSPs. It requires a list of user descriptors and activity (XTM_USER_ACTIVITY).

XTM_USER_ACTIVITY – list of possible activities

- DELETE
  Deletes user

4. findUser()

This method can be used to search for users. It is necessary to specify search parameters in the XTMFindUserAPI object. This method returns all users with details matching the given criteria. Possible criteria are: list of XTMUserBaseDescriptorAPI objects and list of XTM_USER_ROLE_API enumeration types.
By default language combinations for linguists are not provided. They can be obtained by setting the fetchLanguages field in options.

XTM_USER_ROLE_API – list of possible roles

- PROJECT_MANAGER
- TRANSLATOR
- REVIEWER
- CORRECTOR
- TERMINOLOGY_EXPERT
- TM_EXPERT
- LSP

5. createCustomerPM()

A Customer Project Manager can be created by specifying the XTMCustomerPMAPI object. The information is divided into the following groups:

- user details (contains also: address details and instant messaging identifiers)
- customer descriptor
- terminology rights
- TM rights

Most of the fields are not mandatory however the following must be specified:

- user details
    - username
    - password
    - first name
    - last name
    - e-mail address
- customer descriptor

TM rights can be set by using:

- tmAccessRights field – set of TM rights (see TmRightAPI)
- tmAccess – if true then all TM rights will be added

CustomerAdminRoleAPI - possible customer project manager roles

```
/** Manager role. */
MANAGER,
/** Viewer role. */
VIEWER;
```

WorkflowAccessLevelAPI - possible workflow access levels

```
        /** Trusted level. */
        TRUSTED,

        /** Standard level. */
        STANDARD,

        /** Minimal level. */
        MINIMAL;
```

TermRightAPI - possible terminology rights

```
    /** Export right. */
    EXPORT,

    /** Import right. */
    IMPORT,

    /** Add right. */
    ADD,

    /** View right. */
    VIEW,

    /** Modify right. DEPRECATED. */
    MODIFY,

    /** Update and approve right. */
    UPDATE_APPROVE,

    /** Delete right. */
    DELETE,

    /** Suggest right. */
    SUGGEST;
```

TmRightAPI - possible TM rights

```
    /** Export right. */
    EXPORT_TM,

    /** Import right. */
    IMPORT_TM,

    /** View right. */
    VIEW_TM,

    /** Update and delete right. */
    MODIFY_TM,

    /** Update right. */
    MODIFY_UPDATE_ONLY_TM,

    /** Delete right. */
    DELETE_TM;
```

3.3 Project creation

There are four ways of creating a project:

1. createProjectURL()
   Based on file URL, requires XTMProjectURLAPI object

2. createProjectMTOM()
   Based on MTOM objects, requires XTMProjectMTOMAPI object

3. createProjectForPMURL()
   Creates a project with additional settings only available to in-house Project Managers. Based on file URL, requires XTMProjectForPMURLAPI.

4. createProjectForPMMTOM()
   Creates a project with additional settings only available to in-house Project Managers. Based on MTOM or BASE64 depending on the chosen web service, requires XTMProjectForPMMTOMAPI.

Methods 3 and 4 are only available to in-house project managers. They include all the settings available in methods in 1 and 2 but also permit additional settings described at the end of this section.

XTMProjectBaseAPI - the base project information:

```
/** Name of the project. (Optional - if not present will be auto generated) */
String name;

/** Source language. (Mandatory.) */
LANGUAGE_CODE sourceLanguage;

/** Target languages. (Mandatory, at least one must be present) */
List<LANGUAGE_CODE> targetLanguages;

/** Customer descriptor. */
XTMCustomerDescriptorAPI customer;

/** Due date. (Optional, if missing the current date is assumed) */
Date dueDate;

/** Domain descriptor. */
XTMDomainDescriptorAPI domain;

/** Workflow  descriptor. (Mandatory) */
XTMWorkflowDescriptorAPI workflow;

/** Workflow  descriptor for non-analyzable files (Optional). */
XTMWorkflowDescriptorAPI workflowForNonAnalyzableFiles;

/** External project descriptor. (Optional) */
XTMExternalProjectDescriptorAPI externalDescriptor;

/** Templates can be used to create a project. If any project field is not specified in this object, then
if it is available it will be taken from the template. (Optional) */
XTMTemplateDescriptorAPI template;
```

```
/** Analysis template descriptor. (Optional) */
XTMAnalysisTemplateDescriptorAPI analysisTemplate;
```

XTMProjectAPI extends XTMProjectBaseAPI – the extended project information:

```
/** Additional description. (Optional) */
String description;

/** The reference Id. (Optional) */
String referenceId;

/** Info if the linguists should have possibility to use google machine translation help.  (Optional) */
Boolean useGoogleMachineTranslation;

/** Info if not approved translation memory should be used. (Optional) */
Boolean usesNotApprovedTranslationMemory;

/** Allows editing ICE matches. (Optional) */
Boolean allowEditingOfICEMatches;
/** Specifies in which steps non translatables should be marked as done. (Optional) */
WorkflowStepUsageEnum nonTransAsDONE;

/** Specifies in which steps leverages should be marked as done. (Optional) */
WorkflowStepUsageEnum leveragesAsDONE;

/** Specifies in which steps matches from not approved TM should be marked as done. (Optional)
*/
WorkflowStepUsageEnum notApprovedLeveragesAsDONE;

/** Specifies in which steps exact matches from not approved TM should be marked as done.
(Optional) */
WorkflowStepUsageEnum notApprovedExactMatchesAsDONE;

/** Possibility to assign the specified project manager (optional). Cannot be used with
preprocessing (newSourceLanguage). */
XTMUserBaseDescriptorAPI projectManager;

/** Possibility to join files (Optional). Cannot be used with preprocessing (newSourceLanguage).*/
XTM_FILE_PROCESS fileProcessType;

/** Possibility to set TM penalty profile. */
XTMPenaltyProfileDescriptorAPI tmPenaltyProfile;

/** Possibility to define list of TM tags. */
List<XTMTagDescriptorAPI> tmTags;

/** Possibility to set Term penalty profile. */
XTMPenaltyProfileDescriptorAPI termPenaltyProfile;

/** Possibility to define list of Term tags. */
List<XTMTagDescriptorAPI> termTags;

/** Specifies the project manager who manages the customer. */
XTMProjectManagerTypeEnum projectManagerType;

/** Possibility to assign links which should be called on specific project actions. */
XTMProjectCallbackAPI projectCallback;
```

```
    /** Matching options. */
    XTMMatchingAPI matchingOptions;

    /** New source language code (pivot language code). Cannot be used with the join files option.
    (fileProcessType - JOIN) */
    LANGUAGE_CODE newSourceLanguage;

    /** New source workflow descriptor. */
    XTMWorkflowDescriptorAPI newSourceWorkflow;

    /** Possibility to use not approved terms in decoration process. */
    XTMTermAllowNotApproveEnum allowNotApproveTerm;

    /** Possibility to disable terminology decoration (by default this is enabled). */
    Boolean useTerminologyDecoration;

    /** Possibility to automatically finish steps where all segments are green/done. */
    XTMAutoWorkflowFinishEnum autoWorkflowFinish;

    /** Possibility to set project custom fields. */
    List<XTMCustomFieldAPI> projectCustomFields;
```

XTMProjectURLAPI extends XTMProjectAPI:

```
    /** Files requiring translation. (Optional. If not set, then a project without files for translation will be
    created.) */
    List<XTMFileURLAPI> translationFiles;

    /** Reference Material files. (Optional) */
    List<XTMFIleURLAPI> materialFiles;
```

XTMProjectMTOMAPI extends XTMProjectAPI:

```
    /** Files requiring translation (Optional. If not set, then a project without files for translation will be
    created.) */
    List<XTMFileMTOMAPI> translationFiles;

    /** Reference Material files. (Optional)*/
    List<XTMFileMTOMAPI> materialFiles;
```

XTMFileURLAPI and XTMFileMTOMAPI extend XTMFileAPI which contains:

```
    /** File name. */
    String filename;

    /** External job descriptors. It maps external ID to the combination of the source file and language
    code. */
    Map<LANGUAGE_CODE, XTMExternalJobDescriptorAPI> externalDescriptors;

    /** List of target languages for this file. If not specified, the target languages for the project will be
    used. */
    List<LANGUAGE_CODE> targetLanguages;
```

Example:

File for translation: simple.txt
Target languages: pl_PL, ru_RU

In this case 2 jobs will be created in XTM. External descriptors create the possibility to assign an external ID for each job (file with language pair).

XTMExternalProjectDescriptorAPI: integrationId

The external project descriptor needs to be specified to use the project integration ID in other methods instead of the project ID from XTM.

XTMExternalJobDescriptorAPI: integrationId
The external job descriptor needs to be specified to use the job integration ID in other methods instead of the job ID from XTM.

XTMProjectCallbackAPI: projectFinishedCallback, jobFinishedCallback, analysisFinishedCallback and workflowTransitionCallback

Includes URLs which should be called on specific project actions:

- projectFinishedCallback
  This link is called when the project is finished. Additionally, the xtmProjectId and xtmCustomerId parameters are added to the URL.
- jobFinishedCallback
  This link is called when the job is finished. Additionally, the xtmProjectId, xtmJobId and xtmCustomerId parameters are added to the URL.
- analysisFinishedCallback
  This link is called when the project analysis has finished. The callback includes information about the project and jobs sent in JSON format. Additionally, the xtmProjectId and xtmCustomerId parameters are added to the URL.
- workflowTransitionCallback
  This link is called when the workflow event has occurred. Each event in the workflow will generate a notification with all the details about the tasks in JSON format. Additionally, the xtmProjectId and xtmCustomerId parameters are added to the URL.
  - Possible workflow tasks
    - ACTIVE – task has been activated
    - FINISHED – task has been finished
    - DEACTIVATED – task has been deactivated
    - REASSIGNED – task has been reassigned to a different user
    - OPENED – task has been reopened
    - OPENED_BUT_WAITING – task is reopened but currently inactive while waiting for other bundles
    - REMOVED – task has been removed
    - DUE_DATE_CHANGED – due date in task has been changed
  - Workflow task includes information about
    - file
    - step
    - bundle
    - target language

XTMMatchingAPI: fuzzySearch, fuzzyC1Visibility, fuzzyC2Visibility, fuzzyC3Visibility

- fuzzySearch
  Specifies when to search for fuzzy matches

(XTMMatchingFuzzySearchEnum)

- fuzzyC1Visibility, fuzzyC2Visibility, fuzzyC3Visibility
  Specifies when fuzzy matches are visible
  (XTMMatchingFuzzyVisibilityEnum)

fuzzyC1Visibility = 95 – 99%
fuzzyC2Visibility = 85 – 94%
fuzzyC3Visibility = 75 – 84%

Each creation method returns a XTMProjectResponseAPI object which includes:

- project descriptor
- name
- jobs – list of XTMJobResponseAPI objects which contain:
- job descriptor
- file name
- source language
- target language

XTM_WORKFLOWS - possible XTM general Workflows

```
/** Define for 'translate'. */
TRANSLATE,

/** Define for 'translate # review'. */
TRANSLATE_F_REVIEW,

/** Define for 'translate # review # review'. */
TRANSLATE_F_REVIEW_F_REVIEW,

/** Define for 'translate -> correct'. */
TRANSLATE_P_CORRECT,

/** Define for 'translate -> correct -> correct'. */
TRANSLATE_P_CORRECT_P_CORRECT,

/** Define for 'translate , correct'. */
TRANSLATE_CORRECT,

/** Define for 'translate , review'. */
TRANSLATE_REVIEW,

/** Define for 'translate , review -> review'. */
TRANSLATE_REVIEW_P_REVIEW,

/** Define for 'translate , correct -> review'. */
TRANSLATE_CORRECT_P_REVIEW,

/** Define for 'translate # correct'. */
TRANSLATE_F_CORRECT,

/** Define for 'translate -> correct # review'. */
TRANSLATE_P_CORRECT_F_REVIEW;

/** Define for 'translate # correct # review'. */
TRANSLATE_F_CORRECT_F_REVIEW;
```

WorkflowStepUsageEnum - possible steps where leverages and non translatables can be marked as done

```
/** None. */
NONE,
/** In all steps. */
ALL_STEPS,
/** Only in the first step. */
FIRST_STEP

/** All steps except the last. */
NOT_LAST_STEP;

XTM_FILE_PROCESS – possible ways of file processing

/** Join files. */
JOIN
```

XTMProjectManagerTypeEnum – possible ways of project manager assignment

```
/** From customer. */
FROM_CUSTOMER;
```

XTMMatchingFuzzySearchEnum – criteria for searching fuzzy matches

```
/** Always search for fuzzy matches. */
ALWAYS,

/** Search for fuzzy matches even if ICE match found. */
EVEN_ICE_FOUND,

/** Search for fuzzy matches even if 100% matches found. */
EVEN_LEVERAGED_FOUND,

/** Never search for fuzzy matches. */
NEVER
```

XTMMatchingFuzzyVisiibilityEnum – options for fuzzy match visibility

```
/** Always show fuzzy matches. */
YES,

/** Only if its approval status is higher than the ICE or leveraged match. */
ONLY_IF_ITS_APPROVAL_STATUS_IS_HIGHER_THAN_THE_ICE_OR_LEVERAGED_MATCH,

/** Only if its XLIFF:doc status is higher than the ICE or leveraged match. */
ONLY_IF_ITS_XLIFFDOC_STATUS_IS_HIGHER_THAN_THE_ICE_OR_LEVERAGED_MATCH,

/** Only if its approval and XLIFF:doc status are higher than the ICE or leveraged match. */
ONLY_IF_ITS_APPROVAL_AND_XLIFFDOC_STATUS_ARE_HIGHER_THAN_THE_ICE_OR_LEVERAGED_MATCH,
/** Do not show fuzzy matches. */
NO;
```

XTMTermAllowNotApproveEnum – options for using not approved terms in decoration process

```
/** Not approved terms will be used. */
ALLOW,

/** Not approved terms will not be used. */
DISALLOW,
```

Forbidden characters in the project name
A project name cannot contain any of the following characters: \, /, :, *, ?, “, <, >, |

Special characters in the file URL
If a translation file URL contains one of the specific characters below, then an additional URL transformation is required to make the file available for the Web Service. The table below shows how specific characters should be processed.

Character	Encoded character	URL example
	%20	http://test%20.xml
`	%60	http://test%60.xml
~	~	http://test~.xml
!	!	http://test!.xml
@	@	http://test@.xml
#	%23	http://test%23.xml
$	$	http://test$.xml
%	%25	http://test%25.xml
^	%5E	http://test%5E.xml
&	%26	http://test%26.xml
(	(	http://test(.xml
)	)	http://test).xml
-	-	http://test-.xml
_	_	http://test_.xml
=	=	http://test=.xml
+	+	http://test+.xml
{	%7B	http://test%7B.xml
}	%7D	http://test%7D.xml
[	%5B	http://test%5B.xml
]	%5D	http://test%5D.xml
;	;	http://test;.xml
‘	‘	http://test'.xml
,	,	http://test,.xml
.	.	http://test..xml
*	Forbidden	
:	Forbidden	
“	Forbidden	
<	Forbidden	
>	Forbidden	
?	Forbidden	
/	Forbidden	
|	Forbidden	
\	Forbidden	

Additional settings for methods 3 and 4:

- A list of TM Customers (XTMCustomerDescriptorAPI objects)
  This provides the possibility to specify that the project should use the TM of multiple customers. This option requires customer descriptors.

3.4 Project management

1. assignLinguistToProject()

   Assigns linguists or LSPs to all jobs (all bundles, all files and all languages) for a specific workflow step in the project and returns the result of the assignment operation. It requires project descriptor and a list of XTMStepLinguistAssignmentAPI objects.The XTMStepLinguistAssignmentAPI object includes workflow step descriptor and user descriptor.

2. assignLinguistToJob()

   Assigns linguists or LSPs to the specified jobs and returns the result of the assignment operation. It requires a list of XTMJobLinguistAssignmentAPI objects.The XTMJobLinguistAssignmentAPI object includes job descriptor and a list of XTMStepLinguistAssignmentAPI objects which are described in the point 1.

3. startProject()

   Starts the given projects. It requires a list of projects descriptors.
   All jobs in the projects are started and linguists receive emails about their tasks.

4. findProject()

   This method can be used to search for projects. It is necessary to specify search parameters in the XTMFilterProjectAPI object. It returns all projects matching the given criteria with details. There aretwo additional settings in the method's object XTMFindProjectOptionsAPI. The projectCreator parameter filters the projects by the user who created them. The pagination parameter splits the returned project listing into pages and displays only the requested page. The number of projects listed on a page can be specified using the pageSize parameter. Pagination is optional, and when it is not used, all projects are displayed on a single page. The user is specified in the LoginAPI object (XTMProjectCreatorEnum).

5. updateProject()

   This method can be used to update project details. It requires a list of XTMUpdateProjectAPI objects specifying the projects to update.

   The XTMUpdateProjectAPI object includes project descriptor, fields to update (name, description and domain descriptor), project custom fields and project translation properties to update. When the project domain is updated, estimates have to be recalculated. Custom fields can be set or changed for the project.

   The XTMUpdateProjectTranslationPropertiesAPI object includes translation properties which can be updated:

   - allow editing ICE matches

6. updateProjectActivity()

   This method can be used to delete, activate, archive or reanalyze specified projects. It requires a list of project descriptors and activity. There is an additional setting in the method options which filters the projects by the user who created them. The user is specified in the LoginAPI object (XTMProjectCreatorEnum).

7. updateJobActivity()

   This method can be used to cancel specific jobs. It requires a list of job descriptors and the activity to be set (available job activity states are listed in the XTMJobActivity object). Cancelled jobs are no longer processed but they are not removed from projects.

8. updateProjectWorkflow()

   This method can be used to change the workflow for the specified project. It requires the project descriptor and the XTMUpdateWorkflowAPI object where the workflow can be defined using the XTMWorkflowDescriptorAPI object or definitions of workflow steps (the list of XTMUpdateWorkflowStepAPI objects).

The XTMUpdateWorkflowStepAPI object includes the step descriptor, role, forward blocking type and TM approval step.

There are two algorithms for updating the workflow which can be specified by XTM_WORKFLOW_MATCH enum:

- MATCH_NAMES - XTM tries to find the existing workflow step with the same name and updates the properties. Changing the step role will clear any assigned linguists.

- NO_MATCH – XTM deletes the current workflow steps and creates new ones.

9. updateJobGroupWorkflow()
   This method can be used to change the workflow for a specified group of project jobs. It requires the XTMJobGroupDescriptorAPI object which can be used to define a group of project jobs in the following ways:
   - jobDescriptors – a list of project jobs from a single project
   - projectDescriptor – all project jobs from a project
   - projectDescriptor and projectTargetLanguages – a list of project jobs from a project for specific target languages

   The workflow definition and possible workflow update algorithms are described under the updateProjectWorkflow() method.

10. updateTargetLanguages()
    This method can be used to change target languages for the specified project. It requires the XTMUpdateTargetLanguagesAPI object which includes the project descriptor and objects that define changes in the target languages:
    - XTMAddTargetLanguagesAPI – languages to add,
    - XTMDeleteTargetLanguagesAPI – languages to delete.

    This operation cannot be performed when the project is being analysed.

11. updateTranslationFilesMTOM()
    This method can be used to add, update or delete source files in a project using MTOM or BASE64 depending on the chosen webservice. It requires the project descriptor and a list of files. Source files can be added or updated only for all languages. By specifying target languages, in the xtmDeleteTranslationFilesAPI object, source files can be deleted only for a selection of languages. This operation cannot be performed when the project is inactive or being analysed.

12. updateTranslationFilesURL()
    This method can be used to add, update or delete source files in the project based on the file URL. It requires the project descriptor and a list of files. Source files can be added or updated only for all languages. By specifying target languages, in the xtmDeleteTranslationFilesAPI object, source files can be deleted only for a selection of languages. This operation cannot be performed when the project is inactive or being analysed.

13. moveJobWorkflow()
    This method can be used to move the workflow forward or backward or restart a failed automatic step for specific jobs. It requires XTMMoveJobWorkflowAPI object, which consists of a list of XTMJobWorkflowMoveAPI objects.

    The XTMJobWorkflowMoveAPI object includes the job descriptor and the field that defines the type of movement that should be made (XTM_WORKFLOW_MOVE).

    Additional settings:

    - concurrentStepsAsOne – defines if concurrent steps should be treated as one.

    - mailing – possibility to disable email notifications (XTMMoveWorkflowMailing)

14. moveProjectWorkflow()

    This method can be used to move the workflow forwards or backwards or restart a failed automatic step for the specific project. It requires the XTMMoveProjectWorkflowAPI object which includes the project descriptor and the field that defines the type of movement that should be made (XTM_WORKFLOW_MOVE).

    Additional settings are the same as in point 8.

15. checkProjectAnalysisCompletion()

    Checks if analysis of the project and its jobs is completed. This method requires project descriptor and returns the general project status (XTM_PROJECT_COMPLETION_STATUS) and status for each job (XTM_JOB_COMPLETION_STATUS).

16. checkJobAnalysisCompletion()

    Checks if analysis of the job is completed. This method requires a list of jobs descriptors and returns the status for each job (XTM_JOB_COMPLETION_STATUS).

17. checkProjectCompletion()

    Checks if the project and jobs are completed. This method requires the project descriptor and returns

    - the general project status (XTM_PROJECT_COMPLETION_STATUS),
    - the status for each job (XTM_JOB_COMPLETION_STATUS)
    - the locked status for each job (XTMJobLockingStatusEnum) – provides locked status information for segments locked in a TIPP package that has been downloaded by a user. If the segments are locked the target file cannot be generated or uploaded by anyone else.
    - the status for all workflow steps in the job (XTM_STEP_COMPLETION_STATUS).

the status for all automatic workflow steps in the job (XTM_AUTOSTEP_COMPLETION_STATUS)

18. checkJobCompletion()

    Checks if the job is completed. This method requires a list of the job descriptors and returns

    - the status for each job (XTM_JOB_COMPLETION_STATUS)
    - the locked status for each job (XTMJobLockingStatusEnum)
    - the status for all workflow steps in the job (XTM_STEP_COMPLETION_STATUS)
    - the status for all automatic workflow steps in the job (XTM_AUTOSTEP_COMPLETION_STATUS)

19. updateJobStepProperty()

    This method can be used to update workflow step due dates for specific jobs and set a due date for the entire project.

20. updateProjectStepProperty()

    This method can be used to update workflow step due dates for all jobs in translation into the specified target languages and set a due date for the entire project.

21. updateJobWorkflowActivity()

    This method can be used to finish, reopen or reset the workflow for specific jobs. A workflow activity setting should be set using the XTM_WORKFLOW_ACTIVITY enumeration type.

22. updateProjectWorkflowActivity()

    This method can be used to finish, reopen or reset the workflow for all jobs in the specified projects. A workflow activity setting should be set using the XTM_WORKFLOW_ACTIVITY enumeration type.

23. findAnalysisTemplates()

    This method can be used to search for analysis templates. It returns all global templates and templates for customers from the list that can be specified in the XTMFindAnalysisTemplatesFilterAPI object.

24. findWorkflow()
    This method can be used to search for workflow definitions. It returns active definitions that match the given criteria. Optional search parameters, such as the list of workflow IDs or workflow names can be specified in the XTMFindWorkflowFilterAPI object.

XTM_WORKFLOW_MOVE – list of possible types of workflow moves

```
/** The workflow will be moved forward. */
FORWARD,

/** The workflow will be moved backward. */
BACKWARD;

/** The failed automatic step will be restarted. */
RESTART;
```

XTM_PROJECT_COMPLETION_STATUS – the possible project completion statuses

```
/** All jobs in the given project are still processing. */
IN_PROGRESS,

/** At least one job in the given project is completed. */
PARTIALLY_FINISHED,

/** All jobs in the given project are completed. */
FINISHED;
```

XTM_JOB_COMPLETION_STATUS – the possible job completion statuses

```
/** A job is still processing. */
IN_PROGRESS,

/** A job is completed successfully. */
FINISHED,

/** A job could not be completed. */
ERROR;
```

XTM_STEP_COMPLETION_STATUS – the possible workflow step completion statuses

```
/** A workflow step is still processing. */
IN_PROGRESS,

/** A workflow step is completed successfully. */
FINISHED,

/** A workflow step is not started. */
ERROR;
```

XTM_AUTOSTEP_COMPLETION_STATUS – the possible completion statuses for auto workflow steps

/** An auto workflow step has been completed successfully. */
SUCCESS,

/** An auto workflow step could not be completed. */
ERROR;

XTMJobLockingStatusEnum – the possible job locked statuses

/** A job is locked. */
LOCKED;

XTMProjectCreatorEnum – provides an option for filtering the projects by project creator

/** Projects created by user specified in LoginAPI object. */
LOGIN_API_USER

XTM_WORKFLOW_ACTIVITY – available workflow states
/** Sets the workflow state as finished. */
FINISH,

/** Reopens the workflow. **/
REOPEN,

/** Sets the workflow state as not started. */
RESET

XTMMoveWorkflowMailing – the possible options for email notifications

/** Email notifications will be sent (default value if not set) */
ENABLED,

/** Email notifications will not be sent. **/
DISABLED

updateJobActivity – the possible job activity statuses

/** Cancels processing of a job without removing it from the project. */
CANCEL

3.5 Project templates

1. createTemplate()
   A template can be created by specifying the XTMTemplateAPI object.

   The following fields must be specified:
   - name
   - external descriptor – optional

   XTMExternalTemplateDescriptorAPI: externalId

   External template descriptor needs to be specified to use the template external ID in other methods instead of template ID from XTM.

2. findTemplate()
   This method can be used to search for project templates. It is necessary to specify search parameters in the XTMFindTemplateAPI object. This method returns all project templates with details matching the given criteria. Possible criteria are: list of XTMTemplateDescriptorAPI objects, list of XTMCustomerDescriptorAPI objects, XTM_TEMPLATE_SCOPE_API enumeration type.

   XTM_TEMPLATE_SCOPE_API - list of possible template scopes

   - ALL
   Global templates and templates assigned to customers

   - GLOBAL
   Only global templates

   - CUSTOMERS
   Only templates assigned to customers

3.6 Metrics, Statistics and Costs

1. obtainProjectMetrics()

    Obtains original and up to date metrics for all jobs and general metrics for the whole project for each target language or one specified target language. This method requires a project descriptor and optional target language in the additional method options.

2. obtainJobMetrics()

    Obtains original and up to date metrics for the specified jobs. This method requires a list of jobs descriptors.

3. downloadProjectMetricsURL()

    Downloads the XLS metrics file. This method requires a project descriptor and returns the URL to the metrics zip file.

4. downloadProjectMetricsMTOM()

    Downloads the XLS metrics file. This method requires project descriptor and returns the files as MTOM or BASE64 depending on the chosen web service . Files are packed as a zip file.

5. obtainProjectStatistics()

    Obtains up to date user statistics for all jobs for the whole project for each target language or one specified target language. This method requires a project descriptor and optionally a target language in the additional method options.

6. obtainProjectAllStatistics()

    Obtains up to date statistics aggregated on the following levels: project, languages and users. This method requires a project descriptor and optionally a list of target languages in the additional method options.

7. generateCost()

    Creates a cost with a given calculation source type (CostCalculationSource) for the specified project.

    CostCalculationSource values used for cost calculation:

    - INITIAL_METRICS – metrics calculated during project analysis
    - CURRENT_METRICS – current project metrics
    - STATISTICS_SOURCE – statistics based on the source text
    - STATISTICS_TARGET – statistics based on the target text

    When time tracking is enabled, CostCalculationSource includes more options:

    - TIME_OR_METRICS_INITIAL – time entered or metrics calculated during project analysis
    - TIME_OR_METRICS_CURRENT – time entered for or current project metrics
    - TIME_OR_STATISTICS_SOURCE – time entered or statistics based on the source text
    - TIME_OR_STATISTICS_TARGET – time entered or statistics based on the source text

    Tracked time or time entered manually is used for the calculation only for workflow steps where any time record exists. The cost of other workflow steps is calculated using initial metrics, current metrics, source or target statistics.

    Additional options:
    - missingRatesProvider (XTMCostsMissingRatesEnum) – inserts a zero value for linguists who do not have a correctly configured rate card
    - missingTimeProvider (XTMCostsMissingTimeEnum) – inserts a zero value when the time spent has not been entered

- assignment (XTMCostAssignmentAPI) – calculates a project cost based on a dummy user without changing workflow assignments. This feature can be used for all bundles or bundles without assignments (XTMCostsOverrideAssignmentEnum).

This method returns cost descriptors which should be used in the method described in point 8.

8. obtainCost()
   Obtains cost for specific cost descriptor (XTMCostDescriptorAPI). If the ID in the base cost descriptor is set then this method returns the cost with the given ID. Otherwise if the project descriptor and calculation source are set then this method returns the latest cost for these parameters. Otherwise if only the project descriptor is set then this method returns the latest cost for the project.

9. obtainProjectEstimates()
   Obtains estimates for the specified project. This method requires the project descriptor and returns the price, tax and delivery date. Requested estimates can be configured using the XTMEstimatesUpdateAPI object which includes the workflow, delivery type, delivery service, number of copies, number of pages, buyer country and buyer EU citizenship parameters. The buyer country and buyer EU citizenship parameters are used to determine whether VAT has to be paid. VAT will be included in the estimates when the seller or the buyer resides outside of the EU or when both of them reside in the same EU country.

10. generateProjectsSimilarity()
    Calculates the number of repetitions between two projects. This method requires a list of projects pairs to compare and returns the similarity descriptors which should be used in the method described in point 11.

11. obtainProjectsSimilarity()
    Obtains the number of repetitions between two projects. This method requires a list of similarity descriptors and returns the XTMProjectsSimilarityMetricsAPI object for each similarity descriptor. XTMProjectsSimilarityMetricsAPI contains the number of repetitions between the projects and additionally for the second project the number of exact matches, leverage matches and the total number of words.

12. updateProjectEstimates()
    This method allows you to update the current estimate for the specified project. It requires the project descriptor and the estimate configuration provided in the XTMEstimatesUpdateAPI object (described in point 9).

XTMCostsMissingRatesEnum – possible options when linguists do not have rate cards:

```
/** Generate zero costs. */
ZERO;
```

XTMCostsMissingTimeEnum – possible options when the time spent has not been entered:

```
/** Generate zero costs. */
ZERO;
```

XTMCostsOverrideAssignmentEnum – possible options for using a dummy user in the generateCost() method:

```
/** The dummy user will be used in bundles with missing assignments. */
OVERRIDE_MISSING,

/** The dummy user will be used in all bundles. */
OVERRIDE_ALL;
```

3.7 Files

1. generateJobFile()
   Creates files for the specified jobs. The type of file can be specified by GENERATED_FILE_TYPE. This method requires a list of job descriptors and returns file descriptors which should be used in the methods described in points 2, 3 and 4. In the case of generating XLIFF files there is a possibility to enable or disable populating the target with the source (flag in additional XLIFF options).
   This method also enables the generation of a TIPP file from the requested list of jobs. In this case all the jobs should belong to the one target language, in the same project, otherwise a validation error will be returned. The method returns only one file descriptor which should be used in the methods described in points 6, 7 and 8.

2. checkJobFileCompletion()
   Checks if the given files have been generated. This method requires a list of files descriptors and returns status for each file (XTM_JOB_FILE_COMPLETION_STATUS). If an error occurs, an additional message will be returned.

3. downloadJobFileURL()
   Downloads all files which have been generated. This method requires a list of files descriptors and returns URLs to files.

4. downloadJobFileMTOM()
   Downloads all files which have been generated. This method requires a list of files descriptors and returns the files as MTOM or BASE64 depending on the chosen web service . Files are packed as a zip file.

5. generateProjectFile()
   Creates a file with a given type (GENERATED_PROJECT_FILE_TYPE) for the specified project. This method requires the project descriptor and returns the file descriptor which should be used in the methods described in points 6, 7 and 8.

6. checkProjectFileCompletion()
   Checks if the given project files have been generated. This methods requires a list of file descriptors and returns the status for each file (XTM_PROJECT_FILE_COMPLETION_STATUS). If an error occurs, an additional message will be returned.

7. downloadProjectFileURL()
   Downloads all files which have been generated. This method requires a list of file descriptors and returns URLs to the files.

8. downloadProjectFileMTOM()
   Downloads all files which have been generated. This method requires a list of file descriptors and returns the files as MTOM or BASE64 depending on the chosen web service. Files are packed either as a zip or a TIPP file.

9. uploadProjectFileURL()
   Uploads a file with a given type (XTM_UPLOAD_PROJECT_FILE_TYPE) for the specified project and step descriptor. This method returns the file descriptor which should be used in the method described in point 6. For a TIPP file the target language will be read from the manifest.

10. uploadProjectFileMTOM()
    Uploads a file with a given type (XTM_UPLOAD_PROJECT_FILE_TYPE) for the specified project and step descriptor using MTOM or BASE64 depending on the chosen web service. This method returns the file descriptor which should be used in the method described in point 6. For a TIPP file the target language will be read from the manifest.

11. updateReferenceFilesURL()
    Uploads new reference materials to the project or customer. This method requires the XTMUpdateReferenceFilesURLAPI object which contains a list of files to upload and either the project descriptor or the customer descriptor. If the project descriptor is set then the files will be uploaded to the specified project. Otherwise if the customer descriptor is set then they will be uploaded to the specified customer.

12. updateReferenceFilesMTOM()
    Uploads new reference materials to the project or customer using MTOM or BASE64 depending on the chosen webservice. This method requires the XTMUpdateReferenceFilesMTOMAPI object which contains a list of files to upload and either the project descriptor or the customer descriptor. If the project descriptor is set then the files will be uploaded to the specified project. Otherwise if the customer descriptor is set then they will be uploaded to the specified customer.

13. downloadProjectMTOM()
    Downloads all available target files from a project. This method requires the project descriptor and returns the files as MTOM or BASE64 depending on the chosen web service. Files are packed as a zip file. There is an additional option to download only the latest version of any uploaded target files (XTMDownloadProjectTypeEnum).

14. downloadProjectURL()
    Downloads all available target files from a project. This method requires the project descriptor and returns the URLs to the files. There is an additional option to download only the latest version of any uploaded target files (XTMDownloadProjectTypeEnum).

15. downloadJobMTOM()
    Downloads all available target files for specified jobs. This method requires a list of job descriptors and returns the files as MTOM or BASE64 depending on the chosen web service. Files are packed as a zip file. There is an additional option to download only the latest version of any uploaded target file (XTMDownloadJobTypeEnum).

16. downloadJobURL()
    Downloads all available target files for specified jobs. This method requires a list of job descriptors and returns the URLs to the files. There is an additional option to download only the latest version of any uploaded target file (XTMDownloadJobTypeEnum).

17. downloadReferenceMaterialsMTOM()
    Downloads reference materials for projects or customers. This method requires a project descriptor or a customer descriptor which can be specified in the XTMDownloadReferenceMaterialsAPI object. If a project descriptor is specified then the method will return all reference materials assigned to the project and the reference materials for the customer assigned to this project. In the case of specifying a customer descriptor the method will only return reference materials for the customer. The files are returned as MTOM or BASE64 depending on the chosen web service . Files are packed as a zip file.

18. downloadReferenceMaterialsURL()
    Downloads reference materials for projects or customers. This method requires a project descriptor or a customer descriptor which can be specified in the XTMDownloadReferenceMaterialsAPI object. If a project descriptor is specified then the method will return all reference materials assigned to the project and the reference materials for the customer assigned to this project. In the case of specifying a customer descriptor the method will only return reference materials for the customer. This method returns the URLs to the files.

19. uploadXliffURL()
    Uploads a list of XLIFF files for the given job and step descriptors. Autopopulate in additional options will set default values. This method returns the files descriptors which should be used in the method described in point 3.

20. uploadXliffMTOM()
    Uploads a list of XLIFF files for the given job and step descriptors using MTOM or BASE64 depending on the chosen web service. Autopopulate in additional options will set default values. This method returns the files descriptors which should be used in the method described in point 3.

21. checkUploadXliffCompletion()
    Checks if XLIFF files are uploaded. This method requires a list of files descriptors and returns the status for each XLIFF file (XTM_UPLOAD_XLIFF_COMPLETION_STATUS).

GENERATED_FILE_TYPE - possible file types that can be generated

```
/** Target file. */
TARGET,

/** XLIFF file. */
XLIFF,

/** QA Report. */
QA_REPORT,

/** Html in WYSIWYG format if possible. */
HTML,

/** Html with table. */
HTML_TABLE,

/** Pdf in WYSIWYG format if possible. */
PDF,

/** Pdf with table. */
PDF_TABLE,

/** TIPP file. */
TIPP,

/** Extended html table. */
HTML_EXTENDED_TABLE,

/** Extended pdf table. */
PDF_EXTENDED_TABLE,

/** Target coloured by match rate. */
TARGET_COLOURED_BY_MATCH_RATE,

/** Target coloured by the status. */
TAGET_COLOURED_BY_XLIFF_DOC_STATUS,

/** Coloured html according to the stylesheet. */
HTML_COLOURED,

/** Coloured pdf according to the stylesheet. */
PDF_COLOURED,

/** XLIFF:doc file. */
XLIFF_DOC,
/** LQA report. */
LQA_REPORT,

/** LQA report with a detailed list of the LQA errors. */
LQA_EXTENDED_TABLE_REPORT,

/** Pseudo target file. */
TARGET_PSEUDO,

/** PDF coloured by the XLIFF:doc status. */
PDF_COLOURED_BY_XLIFF_DOC_STATUS,

/** PDF coloured by the match rate. */
PDF_COLOURED_BY_MATCH_RATE;
```

XTM_JOB_FILE_COMPLETION_STATUS - possible job file completion statuses

```
    /** File is still generating. */
    IN_PROGRESS,

    /** File has been generated. */
    FINISHED,

    /** File could not be generated. */
    ERROR;
```

GENERATED_PROJECT_FILE_TYPE – possible project file types which can be generated

```
   /** TIPP file. */
   TIPP

   /** LQA report. */
   LQA_REPORT

   /** LQA report with a detailed list of the LQA errors. */
   LQA_EXTENDED_TABLE_REPORT,

   /** Multi-lingual excel. */
   MULTI_EXCEL

   /** PDF file. */
   PDF

   /** PDF coloured by the match rate. */
   PDF_COLOURED_BY_MATCH_RATE,

   /** PDF coloured by the XLIFF:doc status.  */
   PDF_COLOURED_BY_XLIFF_DOC_STATUS;
```

XTM_PROJECT_FILE_COMPLETION_STATUS – possible project file completion statuses

```
   /** File is still generating. */
   IN_PROGRESS

   /** File has been generated. */
   FINISHED
```

```
/** File could not be generated. */
ERROR
```

XTM_UPLOAD_PROJECT_FILE_TYPE – possible project file types to upload

```
/** TIPP. */
TIPP,

/** A ZIP archive containing preview files. */
PREVIEW_FILES,

/** A ZIP archive containing images relating to segment IDs. */
SEGMENT_ID_FILES,
```

XTM_UPLOAD_XLIFF_COMPLETION_STATUS - possible XLIFF completion statuses

```
/** XLIFF is still uploading. */
IN_PROGRESS,

/** XLIFF upload is completed successfully. */
FINISHED,

/** XLIFF upload could not be completed. */
ERROR,

/** The given XLIFF does not exist. */
NOT_EXIST;
```

3.8 Metadata

These methods allows you to obtain additional information about jobs, projects, XTM system information and supported files.

1. obtainJobExtraInfo()
   Obtains additional identifiers for the given list of jobs. This method requires a list of jobs descriptors and returns the list of XTMJobExtraInfoResponseAPI objects.

2. obtainProjectExtraInfo()
   Obtains additional identifiers for all jobs in the given project. This method requires project descriptor and returns the list of XTMJobExtraInfoResponseAPI objects.

3. getXTMInfo()
   Obtains the base information about XTM. Returns the XTMInfoResponseAPI object.

4. getSupportedFilesInfo()
   Obtains the list of supported and not supported files' extensions. For some not supported files there is a list of alternative extensions which can be used instead of the given file, for example: DOC and DOCX can be used instead of DOCM.

5. obtainCustomFields()
   This method can be used to search for custom fields. It returns all custom fields matching the given criteria and includes the details of the custom fields. Search parameters can be specified in the XTMCustomFieldsFilterAPI object.

6. obtainLanguageCombinations()
   This method can be used to obtain language combinations for a customer. It returns the language combination type (XTMLanguageCombinationType). When the customer uses customised language combinations, the method returns the list of the defined language pairs as well. Language definitions, including language codes and localized language names can be obtained using an additional fetchLanguageDefinitions option.

XTMJobExtraInfoResponseAPI - additional identifiers

```
/** Job descriptor. */
jobDescriptor,

/** ID of the XLIFF file. */
xliffFileId,

/** Project descriptor. */
projectDescriptor,

/** ID of the target language. */
targetLanguageId,

/** Target language code. */
targetLanguageCode,

/** Name of the XLIFF file. */
xliffFileName,

/** ID of the source file. */
originalFileId,

/** Name of the original file. */
originalFileName;
```

XTMInfoResponseAPI – base information about XTM

```
/** Company name. */
companyName,

/** URL to the website. */
website,

/** Path to logo image. */
logo,

/** XTM Version. */
version;
```

XTMLanguageCombinationType – a list of language combination types

```
/** All language combinations; the list of language pairs will not be provided. */
DEFAULT,

/** Language combinations defined by the XTM user. */
CUSTOMISED,
```

3.9 Links to XTM Modules

These methods provide links that allow you to open the different modules of XTM in an iframe.

1. obtainPMProjectEditorLink()

   Returns the URL to the PM Project Editor for the given project. It requires project descriptor and optional settings which can be specified in the PMProjectEditorOptions object. Options allow to set visibility for the following tabs: generals, metrics, statistics, workflow, files, estimates. By default all tabs will be visible.

2. obtainPMTranslationPropsLink()

   Returns the URL to the PM Translation Properties page. This configuration can be opened for the whole system or the given customer. Additional options in PMTranslationPropsOptions object allow to set visibility for the following parts: translation, application options, segment status, machine translation.

   The following data should be filled:

   - configuration level
   - customer descriptor – only when configuration level is set to Customer

   ConfigurationLevel – possible configuration levels

   /** Configuration for Client. */
   CLIENT,

   /** Configuration for Customer. */
   CUSTOMER;

3. obtainXTMEditorLink()

   Returns the XTMEditorURLResponseAPI object that contains a URL to the Editor for the specified job and the project activity status. The Editor can be opened only for active projects. It requires the XTMEditorAPI object.

   The XTMEditorAPI contains the information required to prepare the Editor URL in two ways:

   - based on the job, user and customer from XTM. The following fields should be specified:
     - user descriptor
     - job descriptor
     - customer descriptor
     - user options: role
     - workflow options: currentWorkflowStep, lqaDecisionType (LQADecisionType), manageType (XTMEditorWorkflowManageTypeEnum)
   - based on the XLIFF file, user and customer who do not necessarily exist in XTM. The following fields should be specified:
     - user outer descriptor: id, name, preferredLanguage
     - job outer descriptor: xliffFileId, xliffFilePath, projectId, projectName, targetLanguageId, sourceFilePath, xtmXliff
     - customer outer descriptor: id, name
     - user options: role, terminologyRights (EDITOR_TERMINOLOGY_RIGHTS_API)

  - workflow options: workflowSteps, currentWorkflowStep, manageType (XTMEditorWorkflowManageTypeEnum)

There is an additional option to open XTM Visual Editor instead of XTM Standard Editor. This mode can be enabled by setting field editorMode (EditorModeEnum) in editor options. XTM Visual Editor is available for HTML and XML (with defined XSLT) files only. If the XTM Visual Editor is not available then the XTM Standard Editor will be opened by default.

Field manageType allows to manage workflow steps in the Editor. The following criteria must be met for this to happen:

- jobDescriptor filled with data
- step must exist in workflow
- current bundles must be active

```
XTMEditorWorkflowManageTypeEnum
    /** Allows to finish a step. */
    FINISH;

EditorModeEnum
    /** XTM Standard Editor. */
    STANDARD,
    /** XTM Visual Editor if available. */
    VISUAL;

EDITOR_TERMINOLOGY_RIGHTS_API
   /** Add right. */
   ADD,
   /** Modify right. DEPRECATED. */
   MODIFY,
   /** View right. */
   VIEW,
   /** Import right. */
   IMPORT,
   /** Export right. */
   EXPORT,
   /** Delete right. */
   DELETE,
   /** Update and approve right. */
   UPDATE_APPROVE,
   /** Suggest right. */
   SUGGEST;
```

LQADecisionType

```
/** It is not possible to add or view LQA errors.*/
NO,
/** It is possible to add or view LQA errors. Errors will contain information about the user who carried out the LQA and about the user who performed the translation. */
YES_SAVING_RESULTS_FOR_USER,
/** It is possible to add or view LQA errors. The information about the users involved is not recorded in the system. */
YES_NOT_SAVING_RESULTS_FOR_USER,
/** It is possible to view the existing LQA errors. */
NO_SHOW_EXISTING_ERRORS;
```

4. obtainXTMTermManagerLink()

Returns the URL to the Terminology Manager. It requires the XTMTermManagerAPI object.

The XTMTermManagerAPI contains the information required to prepare the Terminology manager URL in two ways:

- based on the user and customers from XTM. The following fields should be specified:
  - user descriptor
  - customers descriptors
- based on the user and customers who do not necessarily exist in XTM. The following fields should be specified:
  - user outer descriptor: id, name, preferredLanguage
  - customers outer descriptors: id, name
  - user options: termRights

XTM_TERM_MANAGER_RIGHTS_API – possible terminology rights

```
/** Add right. */
ADD,

/** Modify right. DEPRECATED. */
MODIFY,

/** View right. */
VIEW,

/** Import right. */
IMPORT,

/** Export right. */
EXPORT,

/** Delete right. */
DELETE;
/** Update and approve right. */
UPDATE_APPROVE,
/** Suggest right. */
SUGGEST;
```

5. obtainXTMTmManagerLink()

   Returns the URL to the TM Manager. It requires the XTMTmManagerAPI object.

   The XTMTmManagerAPI contains the information required to prepare the URL to the TM manager in two ways:

   - based on user and customers from XTM. The following fields should be specified:
     - user descriptor
     - customers descriptors
   - based on user and customers who do not have to exist in XTM. The following fields should be specified:
     - user outer descriptor: id, name, preferred language
     - customers outer descriptors: id, name
     - user options: tmRights

   XTM_TM_MANAGER_RIGHTS_API – possible TM rights

   ```
   /** Update and delete right. */
   MODIFY,

   /** View right. */
   VIEW,

   /** Import right. */
   IMPORT,

   /** Export right. */
   EXPORT,

   /* Update right. /
   MODIFY_UPDATE_ONLY,

   /** Delete right.*/
   DELETE;
   ```

6. obtainPMTemplateEditorLink()

   Returns the URL to the PM Template Editor for the given template. It requires the template descriptor and optional settings which can be specified in the PMTemplateEditorOptions object. Options allow users to set the visibility for the following groups: project settings, customer settings, translation settings, machine translation settings. By default all parts are visible.

7. checkUserLogin()

   Checks if a user with the given login credentials can log in to XTM. XTMUserLoginAPI object contains user login credentials: id, username, password and optionally the XTM instance descriptor (XTMInstanceDescriptorAPI). If XTMInstanceDescriptorAPI is specified then the user will be checked into this XTM account.

   The XTMInstanceDescriptorAPI object contains information about the XTM account: id, name and instance type (XTMInstanceTypeEnum).

This method returns one of the following result (XTM_USER_LOGIN_RESULT_API):

```
/** User can log in. */
SUCCESS,

/** Invalid user login credentials. */
INVALID_LOGIN_CREDENTIALS,

/** Too many users are currently logged. */
TOO_MANY_USERS,

/** Unable to check the licence for the specified client. */
INVALID_LICENSE,

/** Password is correct but has expired. */
PASSWORD_EXPIRED,

/** Max number of login attempts has been exceeded. */
TOO_MANY_INVALID_LOGIN_ATTEMPTS,

/** User account has expired. */
USER_ACCOUNT_EXPIRED;
```

XTMInstanceTypeEnum – possible XTM instance types

```
/** LSP. */
LSP;
```

8. obtainXTMProjectManagerLink()
   Returns the URL to the Project Manager for the specified user. It requires the XTMProjectManagerAPI object which contains XTMUserLoginAPI with user login credentials (described in point 7).

   By default, when the window is opened via the API, the logout button in the top-right hand corner is not visible. It is possible to toggle the visibility of the logout button. When fullWindowMode is set to true (false being the default value) the logout button is visible.

9. obtainPMConcordanceLink()
   Returns the URL to the Concordance tab. It requires the XTMConcordanceAPI object.

   The XTMConcordanceAPI object contains the information required to open Concordance in a new window or tab. There are two ways of defining information displayed:

   - based on an XTM user: customers and language combinations are taken from theuser settings. The following fields should be specified:
     - user descriptor
   - based on customers and language combinations. The following fields should be specified:
     - customersDescriptors
     - languageCombinations

3.10 TM Management

1. importTMURL()
   This method allows you to import TMX or XLIFF files for a customer and language combination. You can also specify the TM tags that should be attached to the TM and set the status of the imported segments (SegmentStatusEnum). This method returns the file descriptors which should be used in the method described in point 3.

2. importTMMTOM()
   This method allows you to import TMX or XLIFF files for a customer and language combination using either MTOM or BASE64 depending on the chosen web service. You can also specify TM tags that should be attached to the TM and set the status of the imported segments (SegmentStatusEnum). This method returns the file descriptors which should be used in the method described in point 3.

3. checkTMCompletion()
   This method can be used to check if the importing or exporting process has finished. This method requires a list of file descriptors and returns the status of each file (XTM_TM_COMPLETION_STATUS).

4. exportTM()
   This method allows you to export TMX files for specified customers. You can also specify the project, source language, target language and TM status. This method returns the file descriptors which should be used in the method described in point 3. There are additional options to include Reverse Memory and to choose the exported file type (XTM_TM_FILETYPE).

5. downloadTMURL()
   Downloads the exported TMX file. This method requires a file descriptor and returns the URL to the TMX zip file.

6. downloadTMMTOM()
   Downloads the exported TMX file. This method requires a file descriptor and returns the files as MTOM or BASE64 format depending on the chosen web service. Files are packed as a zip file.

7. findConcordance()
   This method allows to search concordance. It requires search parameters which can be defined in the XTMConcordanceFilterAPI object and returns a list of segments that match the search criteria, each represented by the XTMTmSegmentAPI object. The required parameters in the XTMConcordanceFilterAPI are:
   - languageCombination or baseProjectDescriptor
     - languageCombination – source and target language
     - baseProjectDescriptor – this contains the “uuid” field which is assigned to the TIPP package and is used to find the base project. The base project will be used to get information about the customer, source and target language.
   - phrase – the phrase to search
   - searchType – this defines whether the search will take place in the source or target segment (XTMConcordanceSearchTypeEnum)

   The optional parameters in the XTMConcordanceFilterAPI are:
   - checkLanguageVariants – set to define language variants for the search
   - customers – set to define list of customers
   - customerSearchType – set to define whether the search will take place for all customers (XTMConcordanceSearcheCustomerTypeEnum)
   - exactMatch – set only to find exact match
   - reverseMemory – set to use reverse memory

XTMConcordanceSearcheCustomerTypeEnum – type of customers to search in.

```
  /** Search in all customers **/
ALL_CUSTOMERS
```

8. findMatches()

   This method allows to search matches in the TM. It requires search parameters which can be defined in the XTMFindMatchesFilterAPI object and returns a list of matches found by using the given criteria, each represented by the XTMMatchAPI object. In the filter criteria you can specify customer, language combination, context information, source sentence and segment id.

9. deleteTM()

   This method allows to delete TM. It requires search parameters which can be defined in the XTMDeleteTMFilterAPI object and returns the process result. In the filter criteria you can specify customer, project name, source and target language, status, tags and XLIFF:doc statuses.

XTM_TM_COMPLETION_STATUS - possible TM completion statuses

```
/** Import/Export is still processing. */
IN_PROGRESS,

/** Import/Export is completed successfully. */
FINISHED,

/** Import/Export could not be completed. */
ERROR,

/** The given TM file does not exist. */
NOT_EXIST;
```

XTMConcordanceSearchTypeEnum – list of possible types for phrase search

```
/** Phrase will be searched in segment source. */
SOURCE,

/** Phrase will be searched in segment target. */
TARGET;
```

XTM_TM_FILETYPE – list of possible TM file types

```
/** TMX file (default). */
TMX,

/** Excel file. */
XLS;
```

SegmentStatusEnum – the list of possible ways of setting the segment status during a TM import

```
/** TM segment status will be automatically set to approved. */
APPROVED,

/** TM segment status will be automatically set to not approved. */
NOT_APPROVED,

/** TM segment status will be set according to their status in the TM import file and to approved
where the status is not defined. */
FROM_FILE_OR_APPROVE,

/** TM segment status will be set according to their status in the TM import file and to not
approved where the status is not defined. */
FROM_FILE_OR_NOT_APPROVE;
```

3.11 Terminology Management

1. importTermURL()
   This method allows to import XLS, XLSX, TBX, MTF files for a specific customer. Additional options:
   - addToExistingTerms – adds new terms to terms that already exist
   - purgeTerms – deletes old terms and imports new terms for a specific customer

2. importTermMTOM
   This method allows to import XLS, XLSX, TBX, MTF files for a specific customer using either MTOM or BASE64 depending on the chosen web service. Additional options are the same as in importTermURL().

3. exportTerm()
   This method allows you to export terminology for a specified customer. You can also specify the main language, list of translation languages, term status, domain and file type (XTMTermFileTypeEnum). If you export an XLS or XLSX file you can choose which columns to export.

4. checkTermCompletion()
   Checks if import or export is completed. This method requires a list of file descriptors and returns the status for each file (XTMTermCompletionStatusEnum).

5. downloadTermURL()
   Downloads the file with exported terminology. This method requires a list of file descriptors and returns the URLs to the exported files.

6. downloadTermMTOM()
   Downloads the file with exported terminology. This method requires a list of file descriptors and returns the files MTOM or BASE64 depending on the chosen web service. Files are packed as a ZIP file.

XTMTermFileTypeEnum – list of possible Terminology file types

```
/** TBX. */
TBX,

/** MTF. */
MTF,

/** XLS. */
XLS,

/** XLSX. */
XLSX;
```

XTMTermCompletionStatusEnum – list of possible statuses for import/export process

```
/** File is still being imported/exported. */
IN_PROGRESS,

/** Import/Export process has been finished. */
FINISHED,

/** Import/Export process has been finished with an error. */
ERROR,

/** A file with the specified ID does not exist. */
NOT_EXIST,
```

4 API examples - SOAP

4.1 JAVA

Java examples are available from XTM International in a ZIP file which can be easily imported to your Eclipse environment as a new project. The file contains:

- Examples for most important methods
- All the required libraries to run the prepared code

All java files are compiled using Java 8.

To download the files for the customer web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/customer-api-examples.zip

To download the files for the project manager web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/pm-api-examples.zip

To download the libraries for the customer web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/customer-jars.zip

To download the libraries for the project manager web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/pm-jars.zip

4.2 PHP

PHP examples are available from XTM International as a separate PHP file. No additional libraries are required. The file contains:

- Examples for most important methods
- A configuration section

To download the files for the customer web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/XTMAPICustomerSoapExample.php%20example

To download the files for the project manager web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/XTMAPIProjectManagerSoapExample.php%20example

For further information please contact sales@xtm-intl.com

4.3 .NET

.NET examples are available from XTM International in a ZIP file which can be easily opened in your Visual Studio environment as a new project. The file contains:

- Examples for most important methods

To download the files for the customer web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/XTMAPICustomerNetExample.zip

To download the files for the project manager web service click here:

- http://files.xtm-intl.com/webservices/v2/latest/XTMAPIProjectManagerNetExample.zip

5 XTM Connect Portal API

5.1 Overview

This section describes the functionality provided by the XTM Portal web service.

The methods in the XTM Portal web service that are borrowed from the XTM Project Manager web service are:

- createProjectURL
- createProjectMTOM
- checkProjectAnalysisCompletion
- checkUserLogin
- findCustomer
- createCustomer
- createCustomerPM
- findProject
- obtainXTMProjectManagerLink
- obtainProjectMetrics
- obtainProjectEstimates
- updateProjectEstimates
- updateProjectActivity
- copyCustomer

The methods that are unique to the XTM Portal web service are:

- obtainXTMPaymentLink
- updateProjectCustomer

5.2 Methods

1. obtainXTMPaymentLink()

   Returns the URL to the SagePay page with all payment information relating to the specified project. It requires the project descriptor and the user descriptor whose details will be used in SagePay.

2. updateProjectCustomer()

   This method changes the customer from the default customer to a specified customer and also provides the option to change the project creator. It requires the customer descriptor, a list of project descriptors and optionally a user descriptor (the user must be a Project Manager or a Customer PM in order to be assigned as the project creator).

6 API examples JSON

This section describes callbacks which are sent from XTM. Callbacks are sent in JSON format as POST parameters.

Analysis finished callback

For example:

QUERY STRINGS
xtmProjectId: 140667
xtmCustomerId: 6095

FORM VALUES

```
{
  "projectManager": {
    "name": "TestPM",
    "id": 18
  },
  "creator": {
    "name": "TestPM",
    "id": 18
  },
  "activity": "ACTIVE",
  "jobs": [
    {
      "fileName": "simple.xml",
      "targetLanguage": "pl_PL",
      "jobDescriptor": {
        "id": 87173
      },
      "status": "FINISHED"
    }
  ],
  "targetLanguages": [
    "pl_PL"
  ],
  "projectDescriptor": {
    "id": 87167
  },
  "name": "TestProject",
  "tmCustomers": [
    {
      "idStr": "21",
      "name": "TestCustomer",
      "id": 21
    }
  ],
  "sourceLanguage": "en_GB",
  "createDate": {
    "date": 4,
    "hours": 10,
    "seconds": 4,
    "month": 7,
    "nanos": 575000000,
    "timezoneOffset": 0,
    "year": 115,
    "minutes": 0,
    "time": 1438682404575,
    "day": 2
  },
  "customer": {
    "idStr": "21",
    "name": "TestCustomer",
    "id": 21
  },
  "status": "NOT_STARTED"
}
```

Workflow transition callback

QUERY STRINGS
xtmProjectId: 140667
xtmCustomerId: 6095

FORM VALUES

For example:

```
{
  "projectDescriptor": {
    "id": 87167
  },
  "events": [
    {
      "type": "FINISHED",
      "tasks": [
        {
          "currentUser": {
            "name": "TestLinguist",
            "id": 29268,
            "type": "INTERNALLINGUIST"
          },
          "fileName": "sample.xml",
          "targetLanguage": "pl_PL",
          "step": {
            "workflowStepName": "translate1",
            "workflowStep": "TRANSLATE1"
          },

          "job": {

             "id": 547047

           },
          "bundle": {
            "from": 1,
            "to": 70
          }
        }
      ]
    }
  ]
}
```

7 Revision history

Date	Changes
22/11/2018	• Added query values to callback examples and extended analysisFinishedCallback and workflowTransitionCallback descriptions
19/10/2018	• Possibility to generate a new file type and project file type: LQA_EXTENDED_TABLE_REPORT
19/06/2018	• New method in the SOAP API: updateJobActivity
28/03/2018	• New field in XTMFindProjectOptionsAPI: pagination
14/02/2018	• Added a download link to NET examples of project manager web service methods
05/01/2018	• New methods in the SOAP API: findWorkflow(), findAnalysisTemplates(), copyCustomer() • New field in XTMProjectBaseAPI: analysisTemplate • New fields in XTMConcordanceFilterAPI: languageCombination, customers
04/01/2018	• New field in XTMProjectDetailsResponseAPI: newSourceLanguage • Possibility to specify new workflow using XTMWorkflowDescriptorAPI in the updateJobGroupWorkflow() and updateProjectWorkflow() methods • New field in XTMProjectEstimatesResponseAPI: taxRate • New fields in XTMEstimatesUpdateAPI: buyerCountry, buyerEUCitizenship • New field in workflow transition callback JSON: job
03/01/2018	• New fields in XTMJobLinguistAssignmentResponseAPI: assignmentStatus, workflowStepsAssignments • Possibility to disable email notification when using moveJobWorkflow or moveProjectWorkflow • New parameter xtmCustomerId added to the URL for project and job finished callback
25/10/2017	• Possibility to update a project description, domain and custom fields. • Added a warning about projects in analysis for the following methods: updateTargetLanguages(), updateTranslationFilesMTOM(). updateTranslationFilesURL(). • Added a warning about inactive projects for the following methods: updateTranslationFilesMTOM() and updateTranslationFilesURL(). • New field in the XTMEditorURLResponseAPI from obtainXTMEditorLink(): projectActivity. • New field in createProject() REST API method: fileProcessType
20/10/2017	The REST API documentation has been removed from this place. A new REST API has been built. Documentation for the new REST API is available here.
25/08/2017	• Possibility to create a customer or a user with custom fields • New methods in the SOAP API: obtainLanguageCombinations(), updateUser(), updateCustomer() • New fields in XTMUserDetailsResponseAPI: dataFormatter, preferredLanguage, customFields • New fields in XTMCustomerDetailsResponseAPI: customFields

	• New fields in the createProject() REST API method: customerId, callbackAnalysisFinished, callbackJobFinished, callbackProjectFinished, callbackWorkflowTransition • New field in the updateTranslationFiles() REST API method: matchType • Added description to the following methods: updateJobWorkflowActivity(), updateProjectWorkflowActivity(), updateJobStepProperty(), updateProjectStepProperty()
01/03/2017	• Possibility to create a project with custom fields • New methods in the SOAP API: updateJobGroupWorkflow(), obtainCustomFields() • New method in the REST API: getXTMVersion() • New field in XTMProjectDetailsResponseAPI: projectCustomFields • New field in XTMImportTMAPI: segmentImportStatus
16/12/2016	• Possibility to search for customers based on their activity status • New field in XTMEditorWorkflowOptionsAPI: lqaDecisionType • New field in PreviewDTO: showExtendedTable
18/11/2016	• Added .NET examples
26/10/2016	• Possibility to create a project without files for translation • Possibility to authenticate using the *username* in the LoginAPI object (SOAP API) • New *message* field in the XTMJobFileStatusResponseAPI and XTMProjectFileStatusResponseAPI objects • New methods in the SOAP API: obtainPMConcordanceLink()
24/06/2016	• Possibility to generate costs when linguists do not have a rate card, when the time spent has not been entered or the workflow assignments are missing • Possibility to delete old terms before new terms are imported
28/04/2016	• Added description of the following methods: exportTerm(), checkTermCompletion(), downloadTermURL(), downloadTermMTOM(), updateTranslationFiles() • New field in XTMUserAdditionAPI: projectsVisibility • New field in XTMProjectAPI: autoWorkflowFinish
14/03/2016	• Possibility to disable terminology decoration and to use not approved terms while creating a project • New method in SOAP API: deleteTM()
19/02/2016	• Possibility to generate PDF coloured by the XLIFF:doc status and by match rate using generateProjectFile() • New fields in createProject() REST API method: description, referenceId, dueDate, newSourceLanguage, newSourceWorkflow • New method in REST API: updateTranslationFiles() • Terminology rights updated in: createUser(), obtainXTMTermManagerLink(), getEditorURL(), createCustomerPM(), obtainXTMEditorLink() • Possibility to set logout button visibility in obtainXTMProjectManagerLink()
17/11/2015	• New methods: importTermMTOM(), importTermURL() • New field in XTMUserAPI: rolesDescriptors • PDF format available in generateProjectFile() • Possibility to set workflow management in obtainXTMEditorLink()
17/08/2015	• New fields in XTMProjectCallbackAPI: analysisFinishedCallback and workflowTransitionCallback • Added description of the following methods: updateTargetLanguages(), updateTranslationFilesMTOM() and updateTranslationFilesURL().

	• Possibility to search concordance for all customers in findConcordance() method. • API examples JSON section added.
21/06/2015	• New methods in XTM Portal Web Service: obtainXTMProjectManagerLink(), findCustomer() and checkUser() • Added new TM rights in methods: obtainXTMTmManagerLink(), createUser(), • Possibility to include Reverse Memory in exportTM() method • Added currency to the response of obtainProjectEstimates() method
24/03/2015	• New methods: obtainXTMPaymentLink(), updateProjectCustomer(), updateProjectEstimates() • New REST methods: deleteProject(), findProject() • Possibility to search projects by status and to obtain the project payment status in the findProject() method • Possibility to filter projects which are created by the user specified in LoginAPI object in findProject() and updateProjectActivity() methods • XTM Connect Portal API section added • Possibility to set matching options while creating a project • Possibility to configure requested estimate in obtainProjectEstimates() • Information about file names in the response of checkProjectAnalysisCompletion(), checkJobAnalysisCompletion(), checkProjectCompletion() and checkJobCompletion() methods
12/02/2015	• Possibility to populate target with source in generated xliff file • Possibility to set different target languages for different source files, set callback URLs and set the Project Manager who manages the customer, while creating a project • Possibility to generate and download a multi-lingual Excel file for a project • Possibility to generate and download pseudo translation for a job • New methods: findMatches()
04/02/2015	• Document layout improved • REST API section added
18/11/2014	• Possibility to generate and download LQA report for job or project • Possibility to obtain information about password expiration • Possibility to update editable ICE matches in the existing project.
03/11/2014	• Possibility to reanalyze project • Possibility to import TM with selected TM tags • Possibility to open Editor with LQA • Possibility to download only the latest version of any manually uploaded target files in project
06/10/2014	• New methods: updateReferenceFilesURL(), updateReferenceFilesMTOM(), obtainProjectEstimates(), generateProjectsSimilarity(), obtainProjectsSimilarity() • Possibility to generate XLIFF:doc file • Possibility to download only the latest version of any manually uploaded target file • Possibility to use new field called integrationId in XTMProjectDescriptorAPI and XTMJobDescriptorAPI . Deprecated externalId. • Possibility to authenticate subcontractor users using checkUserLogin() or obtainXTMProjectManagerLink() methods • Possibility to grant selected TM rights to Customer PM • Possibility to check the job locked status or the status of an automatic step in the workflow using checkJobCompletion() and checkProjectCompletion() methods

	• Possibility to run the automatic step again using moveJobWorkflow() or moveProjectWorkflow() methods
09/06/2014	• Possibility to set workflow for non-analyzable files while creating a project • New methods: moveJobWorkflow(), moveProjectWorkflow() • Possibility to generate: - extended html or pdf table - coloured html and pdf according to a stylesheet - target coloured by match rate or XLIFF:doc status
14/05/2014	• Added new web service for TM related functionality • New method: findConcordance()
07/04/2014	• Possibility to generate a TIPP file for a requested list of jobs • New methods: uploadProjectFileMTOM(), uploadProjectFileURL()
25/02/2014	• New methods: generateProjectFile(), checkProjectFileCompletion(), downloadProjectFileMTOM(), downloadProjectFileURL(), updateProjectStepProperty(), updateJobStepProperty() • Added third option to send and download files - BASE64 encoded binary information
18/11/2013	• New methods: findCustomer(), findUser(), findTemplate(), checkUserLogin(), obtainXTMProjectManagerLink(), generateCost(), obtainCost() • Possibility to set project workflow by ID • Possibility to use workflow steps by defining ID and order number • Added target language code in XTMJobExtraInfoResponseAPI object • Possibility to set term tag groups and term penalty profiles while creating a project • Possibility to export TM as XLS file
23/08/2013	• Possibility to download reference materials for projects and customers
13/08/2013	• Possibility to use TM penalty profiles and tags while creating a project
02/07/2013	• New methods: importTMMTOM(), importTMURL(), checkTMCompletion(), exportTM(), downloadTMMTOM(), downloadTMURL() • Possibility to join files while creating a project
10/06/2013	• New methods: updateCustomerActivity(), updateTargetLanguages() • Possibility to delete translation files using updateTranslationFiles() method
19/04/2013	• New methods: createTemplate(), obtainPMTemplateEditorLink(), updateUserActivity(), updateTranslationFilesURL(), updateTranslationFilesMTOM() • Possibility to use template while creating a project
09/01/2013	• New method: obtainProjectAllStatistics()
26/11/2012	• Possibility to set exact matches from not approved TM to done while creating a project
26/10/2012	• New methods: findProject() and updateProject() • Possibility to configure email notification about projects • Possibility to set 100% matches from not approved TM to done while creating a project
03/07/2012	• Added description of the following methods: updateProjectWorkflowActivity(), updateJobWorkflowActivity() • Possibility to assign a Project Manager while creating a project • Possibility to check the status of workflow steps using checkJobCompletion() and checkProjectCompletion() methods
16/05/2012	• Added PHP example code and Java examples description • Added description of following methods: updateProjectWorkflow(), obtainXTMEditorLink(), obtainXTMTermManagerLink(), obtainXTMTmManagerLink()

	• Possibility to generate new file types: PDF, HTML
12/04/2012	• Document created