



Desktop Connection API reference guide

Revision 1.0

CONTENTS

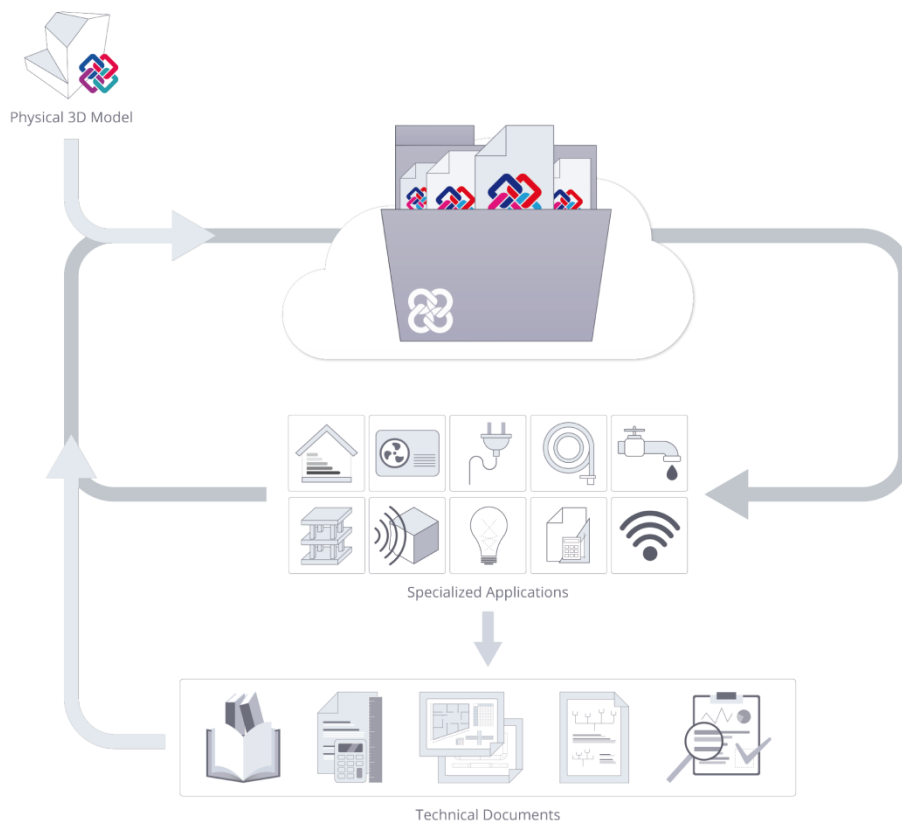
Platform overview	3
The BIM model	4
Files associated to IFC files	5
BIM model updating.....	6
The BIMserver.center synchronizer	7
Integrating the software with the platform	9
Connection API	10
Requirements for the operation of the API	11
Functionality of the API.....	11
Management of the database.....	13
Management of the session.....	13
Management of the linked BIM model	16
Use examples of the API.....	20
Preparation of the API.....	20
Accessing the platform.....	21
Consultation of the access account data	21
Leaving the platform	22
Selection of the BIM project that is related with the work that is going to be carried out or create a new one	22
Selection of an IFC file or associated file as a starting point if the software requires it.	22
Export the results of the work that is carried out to the platform	23
Versioning policy	25
Complementary material.....	25

Platform overview

BIMserver.center is a cloud platform based on Open BIM technology for the collaborative development of software related with architecture, engineering and construction projects.

Open BIM technology:

- Allows users to implant a collaborative and multidisciplinary workflow, with which models can be developed in an open, coordinated and simultaneous manner amongst all the actors that intervene in the development of the project.
- Provides independence with respect to the tools used to develop the project.



For more information, visit www.bimserver.center.

The BIM model

BIMserver.center promotes a workflow in which applications do not exchange their complete models neither in a native nor in a standard format, but only information that is relevant to others. The native data model of the application is always on the user's machine and is not shared via the platform.

For BIMserver.center, the BIM model is a collection of files containing all the information that defines the model up to that moment. The management of the model is carried out via the platform.

From the point of view of its software, the BIM model is a folder that contains a collection of files. Sometimes, the software will select one or several files to carry out its purpose, other times, files will be added to the BIM model to share information with other applications.







The files that make up the model are divided into two categories:

- IFC files: used to exchange information amongst applications.
- Other files: Files that are referenced from IFC files and contain information that is relevant for the model (e.g.: DXF files with structural drawings, PDF files with technical reports, etc.).

The software can share information with (or receive it from) other applications in several ways:

- Using one or more IFC files that contain the information to be shared with (or received from) other applications. It is recommended a single IFC file be generated per application data model.
- Using a specific file format that limits the interoperability amongst applications. In which case, an IFC file must be created (or received) that contains a reference to the user's file or associated files. Any associated file that is generated must be a file with a format that can be read by the platform for previewing reasons (de facto standard: *xls, *.docx or text files: xml, step...*). *If you have any doubts, please contact us.*

The structure of the directory of the BIM model has two levels. The first level contains the IFC files that define the BIM model. If an IFC file has associated files, these will be stored in a folder with the name of the IFC file.

 structural_connections	Files folder	
 architectural_model.ifc	Industry Foundation Classes	15,409 KB
 hvac_floors_1_to_5.ifc	Industry Foundation Classes	97 KB
 rebar_design.ifc	Industry Foundation Classes	28 KB
 structural_3d_model.ifc	Industry Foundation Classes	2,188 KB
 structural_connections.ifc	Industry Foundation Classes	99 KB

Files associated to IFC files

IFC files with references to other documents must meet two conditions:

- There is a directory with the same name as the IFC in the project folder that contains the files to be attached.
- The paths to the attached files that appear in the IFC document are relative to the folder of the BIM project and lead to the documents of the associated directory.

The synchronizer will only upload associated files to the platform if they are referenced by an IFC file that meets the conditions stated above.

For example: the file structural_connections.ifc with an associated file connections.u3d

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('ViewDefinition [CoordinationView]'),'2;1');
FILE_NAME('structural_connections','2017-11-06T16:31:28',(''),(''),'BIMserver.center demo test api','');
FILE_SCHEMA(('IFC4'));
ENDSEC;

DATA;
....
#58= IFCDOCUMENTREFERENCE('.\structural_connections\connections.u3d','Beam – Colum joint',$,'CYPEConnect',$);
....
ENDSEC;
END-ISO-10303-21;
```

If the IFC file is only used as a mechanism to transfer the associated file, then only the basic attributes of the project must be defined. The FILE_DESCRIPTION field in IFC files that are used to encapsulate other files, will be defined as:

```
FILE_DESCRIPTION(('BIMserver.center [IFC Wrapper]'),'2;1')
```

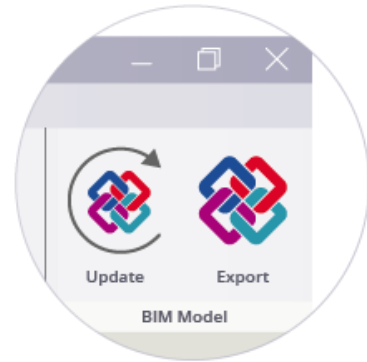
This definition will only be used as a mark to distinguish transport files of the IFC files with contents. The rest of the file will contain the project definition data and references to the associated files.

The connection API includes specific functionality that facilitates the creation of this type of IFC files.

BIM model updating

BIMserver.center promotes a collaborative workflow amongst users, which is achieved by continuously updating the BIM model.

While the synchronization service sees to that the collaborators of a project always have the most updated version, it is up to their software to react against the changes produced in the BIM model that may affect it.



The software can maintain a proactive approach to this issue by implementing a continuous update strategy. The most basic continuous updating strategy should:

- Notify users as soon as possible of any changes in the model
- Assist users to integrate these changes in an intelligent manner

A basic implementation of this strategy requires:

- To store the relationship between the imported BIM entities and the entities of the data model representing them.
- Store the paths relative to the project of the files of the BIM model that have been used.
- Supervision mechanism for changes in the files of the project.

A good continuous updating strategy improves users' BIM experience because they perceive that the information flows from one application to the next, facilitating the work and reducing the possibility of errors arising.

The API connection includes specific functionality to know whether a BIM model file that was used by its application has been modified, which helps to implement this strategy.

The BIMserver.center synchronizer

The synchronizer is the fundamental element that integrates the software with the platform. It consists of two parts:

- The synchronization service
- The user interface to be able to interact with the service.

The synchronization service is a process that is executed in the background as a service of the operating system that:

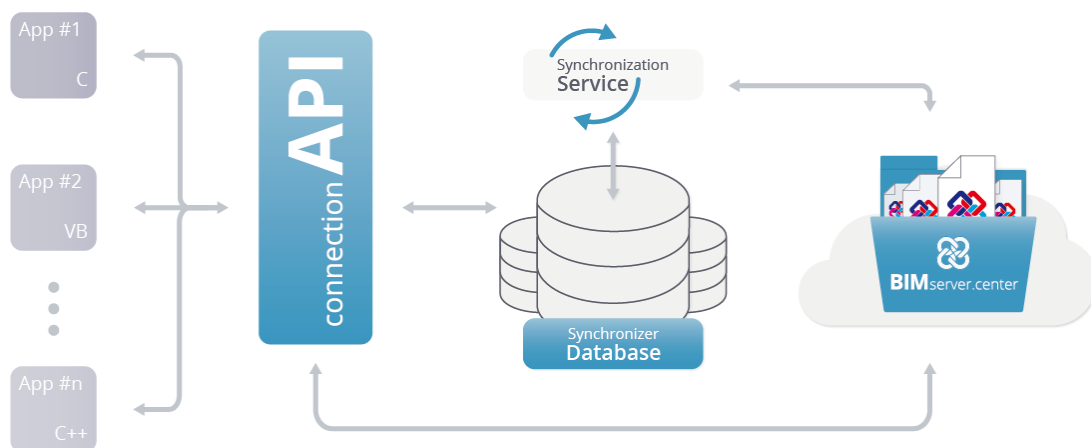
- Keeps users' BIM models up to date
- Loads the results users wish to share with collaborators

The user interface is accessible by users from the task bar and performs the following tasks:

- Monitors the synchronization service
- Manages the access data of the platform
- Selects the projects to synchronize

The term, synchronizer, will be used to refer to both elements. The connection API is integrated in the synchronizer and is distributed jointly with it.

The software will only interact with the platform via the connection API. The synchronizer will manage the projects and their synchronization process (with the network interactions this implies). Regarding the application, only the connection API exists.



The current version of the synchronizer only synchronises projects of a single user. Once the synchronizer has been installed, its application user must access the platform via the synchronizer for the project synchronization process to start.

The connection API provides access to the platform via the synchronizer, so if the user is connected to the platform via the synchronizer, the information of the user that has been introduced will be used by the software.

There is the possibility that the user can connect to the platform using another account from his/her software, in which case, the synchronization of the project will not be effective until the user data is changed in the synchronizer.

A continuous Internet connection is not required to be able to use the software of the platform, however the user must have accessed it at least once and not have disconnected from it.

The synchronizer is currently only available for operating systems of Microsoft Windows XP and higher versions.

Integrating the software with the platform

Even though as software developer you decide the level of integration of your software with the platform, two conditions must always be met:

- Users must have a user account in BIMserver.center
- The BIMserver.center synchronizer must be installed on the user's computer.

The application is considered to have an optimal integration level with the platform if:

- It provides an executable that works in an exclusively integrated manner with the platform. This version will be distributed through the BIMserver.center application store.
- The software installer checks the availability of the BIMserver.center synchronization software on users' computers and installs it if necessary (either because it is not present or because the version is not that which is required).
- Upon starting the application, it checks that the user is connected to the platform. If s/he is not, they will be requested to access it at that time. This way, it does not interrupt users by requesting access data whilst they are operating the software.
- The user account that is being used is visible somewhere in the application.
- Upon creating a new data model, the BIM model user is asked to which model it is to be linked.
- It implements a notification mechanism for changes in the project.
- The software allows users to view information related to other disciplines.

The lowest integration level requires:

- A check of the availability of the BIMserver.center synchronizer and indicate what must be downloaded if it is not installed or updated if the version is not adequate.
- Request the access data to the platform when it must be accessed.

The greater the level of integration of the software with the platform, the more easily the software will be able to send and receive information of the platform, make the most of the advantages it offers and improve the BIM experience of its users.

Connection API

The operability of the API allows users to:

- Configure the language of the forms and replies
- Check the status of the user session
- Access the platform
- Implement its own connection form to the platform
- Manage the database of the API that is linked to the data model
- Obtain the path of the BIM model on the local computer of the user
- Obtain IFC files of the BIM model or add them
- Obtain files associated to the BIM model or add them
- Implement the updating mechanism proposed by BIMserver.center

The interaction with the platform is carried out using a command line tool (**bsapicmdl.exe**), which is installed in the same folder as the synchronizer. The information on the synchronizer can be located in the Windows register (available for 32 and 64 bit applications):

HKEY_LOCAL_MACHINE\SOFTWARE\BIMserver.center\Synchronizer

Where the following fields can be found:

InstallDir: installation directory of the synchronizer. Type [REG_SZ].

Version: {major_version}.{minor_version}.{tweak_version}. Type [REG_SZ]

E.g.:

InstallDir: "C: \Program files\bimserver.center"

Version: "1.0.0"

The functions of the API are executed by calling upon the command line tool with a well defined syntax. The response is stored in a plain UTF-16 text file. The file is composed of one or more lines of text.

The return code is always contained in the first line of the response file. The possible values are:

- YES: Action executed correctly, the following line may contain more return information depending on the API that has been invoked.
- NO: Action executed correctly, but cancelled by the user. There are no more lines in the file.
- BLOCKED_RESOURCE: Action not executed because it has not been possible to access the database of the API. The following line contains a description message of the error.
- ERROR: Action not executed because an error has occurred. The following line contains a description message of the error.

If the response code is YES, the contents of the second and following lines depend on the API that has been invoked. Below are some response file examples:

E.g. API -is_logged. User connected.

YES

E.g. API -select_current_project. The database is blocked.

BLOCKED_RESOURCE

Another application has blocked the access to the database

Requirements for the operation of the API

The API requires a work directory where it will store its own database and the responses to the API calls. The database has a fixed name: **bssession.xml** and contains the information that is required to implement the Open BIM workflow of the BIMserver.center.

It is recommended a specific directory be created for the API and should be contained in the same folder that contains the file/s with your application data model or in the folder where its associated data is stored. This directory must be copied if a copy of the file with the data model is carried out.



Functionality of the API

Calls to the API have the following format:

```
bsapicmdl.exe  
  -lang "<langid>"  
  -wd "<path >"  
  [-trid "<trans_id>"]  
  -<api_call> <api_params>
```

The response is stored in a UTF-16 text file named *<api_call>.txt* in the directory indicated by the *-wd* parameter. If the *-trid* parameter is indicated and the API call accepts it, the response file will have the name *<trans_id>_<api_call>.txt*.

The parameters are divided into two parameter groups: the common parameters and those that depend on the invoked API. The common parameters are listed below:

Parameter	-lang
Use	Configures the language of panels and error messages
Obligatory	No
Possible values	ES (Spanish), EN (English), FR (French), IT (Italian), DE (German)
Example	<i>-lang "FR"</i>

Parameter	-wd
Use	Path of the work directory. This directory contains the session file and text files with the API responses
Obligatory	Yes
Possible values	Any path with write permission
Example	<code>-wd "c: my model model_name.bsapi"</code>

Parameter	-trid
Use	<i>Only available for some APIs</i> Modifies the name of the response file. The response file will be called: <code><trans_id>_api_call.txt</code>
Obligatory	No
Possible values	Any unsigned integer value
Example	<code>-trid "1234"</code>

Parameter	-suffix
Use	Modifies the name of the response file and of the database. The response file will be called: <code>api_call_<suffix>.txt</code> and the database <code>bssession_<suffix>.xml</code>
Obligatory	No
Possible values	Any string that can be part of a file name
Example	<code>-suffix "ab33a0d"</code>

Parameter	-debug
Use	Executes the API in debug mode. Useful to see the command line used to invoke the API.
Obligatory	No
Example	<code>-debug</code>

Management of the database

Function	-create_database
Use	Creation of the initial database
Parameters	None
Concurrent	No
Response	If the operation is correct: YES In any other case: ERROR <error message >
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -create_database

Management of the session

Function	-is_logged
Use	Checks if there is an open user session.
Parameters	None
Concurrent	Yes
Response	If the user is connected YES If the user is not connected NO
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -is_logged

Function	-login
Use	Shows the start dialogue of the session in the platform. If the user has not previously started a session, an Internet connection will be required to complete the session start process.
Parameters	None
Concurrent	No
Response	If the user logs in to the platform YES If the user cancels the operation or is not connected: NO If the database cannot be accessed BLOCKED_RESOURCE <Error message>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -login

Function	-get_logged_user_name
Use	Returns the name with which the user is registered in the platform.
Parameters	None
Concurrent	Yes
Response	<p>If the operation is correct:</p> <p>YES</p> <p><User name></p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -get_logged_username

Function	-get_logged_user_email
Use	Returns the email with which the user is registered in the platform.
Parameters	None
Concurrent	Yes
Response	<p>If the operation is correct:</p> <p>YES</p> <p><user e-mail></p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -get_logged_user_email

Function	-get_logged_user_image
Use	Returns the user image.
Parameters	None
Concurrent	Yes
Response	<p>If the operation is correct:</p> <p>YES</p> <p><path to the user image></p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -get_logged_user_image

Function	-do_login
Use	Carries out the connection process to the platform using the indicated e-mail and password. An Internet connection is required.
Parameters	<user e-mail> <password>
Concurrent	No
Response	<p>If the user is connected to the platform YES</p> <p>If there is an error in the user data ERROR <error message></p> <p>If the database cannot be accessed BLOCKED_RESOURCE <error message></p>
Example	bsapicmdl.exe -lang "EN" -do_login "testuser@bs.com" "1234"

Function	-do_logout
Use	Closes the session that is currently open.
Parameters	No
Concurrent	No
Response	<p>If the user disconnects from the platform YES</p> <p>For any other case ERROR <error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -do_logout

Function	- send_user_recover_password_email
Use	Tells the platform to send the user an email to recover the password. An Internet connection is required.
Parameters	<user e-mail>
Concurrent	No
Response	<p>If the user disconnects from the platform YES</p> <p>For any other case ERROR <error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" - send_user_recover_password_email "testuser@bs.com"

Management of the linked BIM model

Function	-select_current_project
Use	Selects the BIM model to which the data model is to be linked.
Parameters	None
Concurrent	NO
Response	<p>If the operation is correct:</p> <p>YES</p> <p><id of the selected model: unsigned long></p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -select_current_project

Function	-get_current_project_path
Use	Returns the local path of the model in the user's computer.
Parameters	None
Concurrent	Yes
Response	<p>If the operation is correct:</p> <p>YES</p> <p><path of the BIM model folder></p> <p>If the user cancels the operation:</p> <p>NO</p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -select_current_project

Function	-select_ifc_file_from_current_project
Use	Launches a selection dialogue for an IFC file of the BIM model.
Parameters	None
Concurrent	Yes
Response	<p>If the user selects a file:</p> <p>YES</p> <p><local path of the ifc file with respect to the BIM model folder></p> <p>If the user cancels the operation:</p> <p>NO</p> <p>For any other case:</p> <p>ERROR</p> <p><error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -select_ifc_file_from_current_project

Function	-put_file_in_current_project
Use	Adds a new ifc file to the BIM model. If the file already exists, and the user has write permission over it, the file is replaced.
Parameters	<absolute path to the ifc file>
Concurrent	Yes
Response	If the operation ends correctly: YES For any other case: ERROR <error message>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -put_file_in_current_project "c:\my model\structural loads.ifc" bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -put_file_in_current_project "c:\my model\structural loads.loads"

Function	-select_associated_file_from_current_project
Use	Selects a file associated to the BIM model that is referred to from an IFC file of the model.
Parameters	-filter_by_file_extension <file_ext1[;file_ext2; .. file_extn]>
Concurrent	Yes
Response	If the user selects a file: YES <local path of the ifc file with respect to the folder of the bim model> If the user cancels the operation: NO For any other case: ERROR <error message>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -select_associated_file_from_current_project -filter_by_file_extension "txt;pdf"

Function	-put_associated_file_in_current_project
Use	Adds a file associated to the BIM model, which creates an IFC file that references it. If the file already exists, and the user has write permission over it, the file is replaced.
Parameters	<absolute path to the associated file> [-file_desc "File description"] [-app_name "Application name"] [-app_version "Application version"] [-app_desc "Application description"]
Concurrent Response	Yes If the operation ends correctly: YES <local path of the ifc file with respect to the folder of the BIM model> For any other case: ERROR <error message>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" - put_associated_file_in_current_project "c:\my model\pipes.phvac" -file_desc "Defined pipes"

Function	-get_file_path_in_current_project
Use	Having provided a local file path to the BIM model, its absolute path on the user's computer is returned. Use this function to obtain the complete path of a file with the result of the call to: -select_ifc_file_from_current_project -select_associated_file_from_current_project. Each time the absolute path of a file is recovered, it will be noted in the database of its version to be able to implement the updating process. Use the parameter -no_update_db_timestamp to avoid this behaviour.
Parameters	<local path of a file of the model> [-no_update_db_timestamp]
Concurrent Response	Yes If the operation ends correctly: YES <absolute path of the file> For any other case: ERROR <error message>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -get_file_path_in_current_project "structural loads.ifc"

Function	-exists_updated_file_version_current_project
Use	Having provided a local file path to the BIM model, it indicates if there is a more updated version than the last version that was accessed.
Parameters	<local path of a file of the model>
Concurrent	Yes
Response	<p>If there is a more updated version: YES</p> <p>If there is not a more updated version: NO</p> <p>For any other case: ERROR <error message></p>
Example	bsapicmdl.exe -lang "EN" -wd "c:\bsapidemo" -exists_updated_file_version_current_project "structural loads.ifc"

Use examples of the API

A user's normal operation with the platform implies:

1. Preparation of the API.
2. Accessing the platform.
3. Consultation of the access account data.
4. Leaving the platform.
5. The selection of the BIM project that is related with the work that is going to be carried out or create a new one.
6. The selection of an IFC file or associated file as a starting point if the software requires it.
7. Export the results of the work that is carried out to the platform.

Note: The following premises have been adopted for all the cases shown below:

- The work directory is located in c:\model1\model1.bsapi
- The language of use of the API is English
- The user is John Doe, with e-mail: j.doe@acme.org and access code 12345678

Please recall that the active access account of the platform is managed by the synchronizer.

Preparation of the API

The API requires for there to be a directory where the work database and request replies can be stored. In this example, let us assume that the following directory exists: c:\model1\model1.bsapi. If the connection is carried out before selecting the project, it is recommended a directory with a random name be used only to manage the connection.

Once the directory, in which the database of the API is going to be stored, has been prepared, the first call will be:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -create_database
```

In the directory, c:\model1\model1.bsapi, a file called bssession.xml will appear. Do not worry about the contents of the file, they can vary from one version to another. The .xml format is used to guarantee the compatibility of its data models.

If your program allows for the data model to be saved with a different name, do not forget to copy this directory and rename it appropriately.

Accessing the platform

Before performing any interaction with the platform, the software must check if the user is connected to it. The following command can be used to check this:

```
bsapicmdlIn
    -lang "EN"
    -wd "c:\model1\model1.bsapi"
    -is_logged
```

Analyse the file `c:\model1\model1.bsapi\is_logged.txt` to know whether or not the user is connected.

To allow to connect the user to the platform, his/her own form can be implemented or use that provided by the API. For the first case, the user and password fields must be completed. Once the user accepts the form, the `-do_login` function can be used to access the platform:

```
bsapicmdlIn
    -lang "EN"
    -wd "c:\model1\model1.bsapi"
    -do_login "j.doe@ acme.org" "12345678"
```

Analyse the response file `c:\model1\model1.bsapi\do_login.txt`. It is possible to tell the API to request the platform to send a reminder e-mail of the password, if necessary:

```
bsapicmdlIn
    -lang "EN"
    -wd "c:\model1\model1.bsapi"
    -send_user_recover_password_email "j.doe@ acme.org"
```

If you opt to use the functionality of the API, simply use the following command line:

```
bsapicmdlIn
    -lang "EN"
    -wd "c:\model1\model1.bsapi"
    -login
```

The response of the API can be found in the file: `c:\model1\model1.bsapi\login.txt`.

Consultation of the access account data

It is convenient to know that somewhere in the software users can consult what data has been used to connect to the platform. The API provides 3 functions to recover the name, e-mail and user image from the BIMserver.center account.

```
bsapicmdlIn
    -lang "EN"
    -wd "c:\model1\model1.bsapi"
    -get_logged_user_name
```

```
bsapicmdlIn
```

```
-lang "EN"  
-wd "c:\model1\model1.bsapi"  
-get_logged_user_email
```

```
bsapicmdlIn  
-lang "EN"  
-wd "c:\model1\model1.bsapi"  
-get_logged_user_image
```

When the call returns YES, the second line will contain the name, e-mail and path of a file with the user image, respectively.

Leaving the platform

It is not necessary to close the session at the platform, except when users wish to change the account that has been used to connect to the platform. In which case, as software developer you can implement your own solution or use that provided by the API.

For the first case, the following functionality must be used:

```
bsapicmdlIn  
-lang "EN"  
-wd "c:\model1\model1.bsapi"  
-do_logout
```

And analyse the file c:\model1\model1.bsapi\do_logout.txt. For the second case, use the access panel once again to change the user.

Selection of the BIM project that is related with the work that is going to be carried out or create a new one

At some point during the course of the user's task, the user will need to obtain information from the platform or send it. However, prior to this, the user must have selected the project with which s/he wishes to work, using the following functionality:

```
bsapicmdlIn  
-lang "EN"  
-wd "c:\model1\model1.bsapi"  
-select_current_project
```

Analyse the response file; only the first line is of interest.

Selection of an IFC file or associated file as a starting point if the software requires it

Depending on the role the software plays, different information will be required. The connection API offers functionality to manage all use cases. To have control over the work files, the following functionality can be used to obtain the project's path:

```
bsapicmdlIn  
-lang "EN"  
-wd "c:\model1\model1.bsapi"  
-get_current_project_path
```

Analyse the file `c:\model1\model1.bsapi\get_current_project_path.txt`. If the response is YES, the second line of the file indicates the complete path of the project folder.

If only one IFC file has to be obtained, the following API functionality can be used:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -select_ifc_file_from_current_project
```

The user screen displays an IFC file selection dialogue, if the user accepts, the response is YES and the second line of the file contains the path relative to the IFC file in the project folder.

If only one file associated to the IFC file is to be selected, use:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -select_associated_file_from_current_project
  -filter_by_extension "u3d;pdf"
```

The parameter `-filter_by_extension` is obligatory and is composed of a list of extensions separated by the character `;`. If the user accepts, the response is YES and the second line of the file contains the path relative to the IFC file in the project folder.

If the selection functionality provided by the API has been used, the paths of the files are relative to the project, to change them into absolute paths the API must be called upon again:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -get_file_path_in_current_project "local_file_path"
```

If the response is YES, the second line of the file indicates the complete path to the selected file.

Each time this functionality is used, information on the version of that file is stored in the database. This is a small aid to implement the continuous updating mechanism. If you do not wish for this to occur, use the parameter: `-no_update_db_timestamp`.

Export the results of the work that is carried out to the platform

Once the work with the software has concluded, users may wish to share some results with other collaborators. Files can be exported to the platform in 2 ways:

1. By directly accessing the project folder
2. By using the functionality of the API

In the first case, recover the project folder and leave a copy there of the results the user wishes to share with the rest of the platform. Please recall that the synchronizer will only upload IFC files, to upload other type of files, these must associated to the IFC file following the procedure described in the previous sections.

In the second case, the API provides functionality to quickly export one or several files:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -put_file_in_current_project "absolute_file_path"
```

Or directly an associated file, the API creates the IFC casing that is required for the synchronizer to upload it to the platform:

```
bsapicmdlIn
  -lang "EN"
  -wd "c:\model1\model1.bsapi"
  -put_associated_file_in_current_project "c:\steel_connections.u3d"
  -file_desc "Beam-Column connection for Column C1"
  -app_name "SteelTools"
```


Versioning policy

The API follows a numbering system of the following type:

`{major_version}.{minor_version}.{tweak_version}`

- Major versions: the functionality of the API may disappear or change the format of the calls. Backwards compatibility is not guaranteed.
- Minor versions: New functionalities can be added, but neither can functionalities be removed nor the format of the calls changed. Backwards compatibility is guaranteed.
- Tweak versions: New parameters can be added to existing calls and the appearance of dialogues can be changed. Backwards compatibility is guaranteed.

Any major version change will be notified at least 3 months in advance to the registered developers, if this implies a compatibility loss. After that time, the affected functionality may be modified or eliminated in any revision of the synchronizer.

The API version is the same as the synchronizer and is stored in the Windows registry as described previously.

Complementary material

In order to facilitate the use of the API, BIMserver.center has prepared a Visual Studio project with an example implementation of the API for C, C #, Visual Basic .Net and Visual Basic 6 languages.

The link of the project is in the developers section of BIMserver.center.