Данное руководство предназначено для разработчиков программного обеспечения на Windows, желающих использовать внешний API (Application Programming Interface) SimInTech. API SimInTech разработан на основе технологии COM (Component Object Model) с целью обеспечить гибкость и совместимость с различными языками программирования и платформами.

В данном руководстве представлена следующая информация:

- как интегрировать SimInTech в свои проекты, используя его внешний API;
- какие функции доступны через API, как их вызывать и какие параметры они принимают;
- приведен пример разработки консольного приложения, включающий взаимодействие с API SimInTech, с использованием наиболее универсального языка программирования C++ и среды разработки Microsoft Visual Studio 2022.

Руководство поможет эффективно использовать внешний API SimInTech для разработки собственных приложений. При возникновении вопросов или проблем предлагается обратиться в службу технической поддержки.

Технология COM основана на идее, что любое приложение может предоставлять свои функции другим приложениям в виде компонентов. Эти компоненты могут быть использованы другими приложениями для расширения своих возможностей или для взаимодействия друг с другом. Каждый класс COM определяется CLSID, уникальным 128-разрядным GUID, который должен зарегистрировать сервер. COM использует этот CLSID по запросу клиента для связывания определенных данных с библиотекой DLL или EXE, содержащей код, реализующий класс, таким образом, создавая экземпляр объекта. Для клиентов и серверов на одном компьютере CLSID сервера — это все, что требуется клиенту. На каждом компьютере COM поддерживает базу данных (использует системный реестр) всех CLSID для серверов, установленных в системе. Это сопоставление между каждым CLSID и расположением библиотеки DLL или EXE, в которой содержится код для этого CLSID. COM обращается к этой базе данных всякий раз, когда клиент хочет создать экземпляр класса COM и использовать его службы, поэтому клиенту никогда не нужно знать абсолютное расположение кода на компьютере. Регистрация COM-объекта SimInTech выполняется при установке программы автоматически путем выполнения консольной команды:

"$(SIT)\bin\mmain.exe" /regserver

, где *$(SIT)* – директория установки SimInTech.

Описание COM-объекта SimInTech в универсальном формате библиотеки типов в бинарном и текстовом виде для разработчиков стороннего программного обеспечения поставляется вместе с программой в виде файлов:

$(SIT)\source\exe\mmain.tlb

$(SIT)\source\exe\mmain.ridl

Кроме того, для упрощения процесса разработки вместе с программой поставляются заголовочные файлы, содержащие объявление структур, использующихся в API, на двух языках программирования Delphi и C/C++:

$(SIT)\source\exe\mmain_TLB.pas

$(SIT)\source\exe\mmain.hpp

$(SIT)\source\exe\mmain_i.c

Стоит отметить, что наличие заголовочных файлов в составе дистрибутива SimInTech абсолютно не обязательно для разработчика, так как их довольно легко сгенерировать самостоятельно, используя универсальный файл библиотеки типов. Например, при помощи компилятора MIDL для языка C/C++, который распространяется в составе пакета SDK для операционной системы:

call "C:\Program Files\Microsoft Visual Studio\2022\Community\VC\Auxiliary\Build\vcvarsall.bat" x64

midl "C:\SimInTech64\source\exe\mmain.ridl" /out " C:\SimInTech64\source\exe" /h "mmain.hpp"

Зачастую для разработчика крайне необходимо, чтобы приложение, с которым он взаимодействует через API, запускалось по умолчанию в скрытом режиме без пользовательского интерфейса. Этот функционал реализован и в SimInTech. Для его активации следует в файле *$(SIT)\bin\mmain.ini* выставить значение переменной *silentmode=1*. Кроме того, в API предусмотрена функция *SetSilentMode* для включения/выключения данного режима в процессе выполнения программы.

Также реализован функционал, позволяющий стороннему приложению перехватывать лог SimInTech. В параметрах программы на вкладке *Файлы и папки* имеются настройки *Имя Pipe-сервера для передачи журнала* и *Интервал сохранения журнала, мсек*, позволяющие задавать имя Pipe-сервера, к которому SimInTech подключается в качестве клиента и с заданной частотой записывает накопившиеся сообщения лога. Аналогичным функционалом обладают функции API *SetPipeName* и *SetJournalSavePeriod*.

Ниже приведен полный перечень функций (сервисов) и их аргументов, доступных разработчику, использующему API SimInTech:

Имя функции	**Список аргументов**	**Описание**
SetSilentMode	/* [in] */ long Mode	Установить флаг – скрытый режим работы программы без пользовательского интерфейса (silentmode)

Имя функции	Список аргументов	Описание
SetJournalSavePeriod	/* [in] */ long Period	Установить интервал записи (в мс) лога программы в pipe-канал и файл журнала
SetPipeName	/* [in] */ BSTR PipeName	Установить имя pipe-канала для передачи лога программы (SimInTech – клиент)
WaitForAllLoading		Ожидание полной загрузки программы
ProcessAllMessages		Принудительно выполнить все накопившиеся сообщения
SetShutdownOnLastRelease	/* [in] */ long Value	Установить флаг – закрыть приложение при отсоединении от него всех COM-объектов
SetDesktopAsParent	/* [in] */ long Value	Установить флаг независимости окон
SetNoCloseAppFlag	/* [in] */ long Value	Установить флаг запрета закрытия приложения
SetMainFormVisible	/* [in] */ long Value	Установить видимость главной формы приложения
GetProcessID	/* [out] */ unsigned long *PID	Получить идентификатор процесса программы
SetCallbackHandleAndDataID	/* [in] */ unsigned __int64 CallbackHWND /* [in] */ unsigned __int64 CallBackDataId	Установить дескриптор формы и идентификатор данных для колбека отправки информационного сообщения управляющему процессу
GetSystemVariableValue	/* [in] */ BSTR aVarName /* [out] */ VARIANT *aValue	Получить значение внутренней переменной окружения
SetSystemVariable	/* [in] */ BSTR aVarName /* [in] */ BSTR aValue	Установить значение внутренней переменной окружения
GetProjectCount	/* [out] */ long *Count	Получить общее количество загруженных проектов
GetProjectIdByNumber	/* [in] */ long PrjNumber /* [out] */ __int64 *ID	Получить уникальный идентификатор (ссылку) проекта по индексу

Имя функции	Список аргументов	Описание
GetProjectIdByFileName	/* [in] */ BSTR PrjFileName /* [out] */ __int64 *ID	Получить уникальный идентификатор (ссылку) проекта по имени файла
GetProjectByCOMName	/* [in] */ BSTR COMName /* [out] */ __int64 *ProjectId	Получить уникальный идентификатор (ссылку) проекта по COM-идентификатору
GetActiveProject	/* [out] */ __int64 *ProjectId	Получить уникальный идентификатор (ссылку) текущего активного проекта
GetPackCount	/* [out] */ long *Count	Получить общее количество загруженных пакетов проектов
GetPackId	/* [in] */ long PackNumber /* [out] */ __int64 *PackId	Получить уникальный идентификатор (ссылку) пакета проектов по индексу
GetPackIdByFileName	/* [in] */ BSTR PackFileName /* [out] */ __int64 *PackId	Получить уникальный идентификатор (ссылку) проекта по имени файла
PackGetProjCount	/* [in] */ __int64 PackId /* [out] */ long *ProjCount	Получить количество проектов в составе пакета проектов
PackGetProjectIdByIndex	/* [in] */ __int64 PackId /* [in] */ long aIndex /* [out] */ __int64 *ProjectId	Получить уникальный идентификатор (ссылку) проекта в составе пакета проектов по индексу
OpenProject	/* [in] */ BSTR PrjFileName /* [out] */ __int64 *ProjectId	Открыть проект
NewProject	/* [out] */ __int64 *ProjectId	Создать новый пустой проект
OpenTemplate	/* [in] */ BSTR TemplateName /* [out] */ __int64 *ProjectId	Открыть шаблон проекта
CloseProject	/* [in] */ __int64 ProjectId	Закрыть проект

Имя функции	Список аргументов	Описание
SaveProjectBinary	/* [in] */ __int64 PrjId /* [in] */ BSTR FileName	Сохранить проект в бинарный файл
SaveProjectXML	/* [in] */ __int64 PrjId /* [in] */ BSTR FileName	Сохранить проект в XML файл
OpenPack	/* [in] */ BSTR FileName /* [out] */ __int64 *PackId	Открыть пакет проектов
ClosePack	/* [in] */ __int64 PackId	Закрыть пакет проектов
ProjectStart	/* [in] */ __int64 ProjectId	Инициализировать проект
ProjectRun	/* [in] */ __int64 ProjectId	Запустить проект
ProjectStep	/* [in] */ __int64 ProjectId	Выполнить шаг расчета для проекта
ProjectPause	/* [in] */ __int64 ProjectId	Поставить проект на паузу
ProjectStop	/* [in] */ __int64 ProjectId	Остановить проект
RunTo	/* [in] */ __int64 ProjectId /* [in] */ double TargetTime /* [out] */ __int64 *Result	Запустить расчет проекта до заданного времени
WaitForTime	/* [in] */ __int64 ProjectId /* [in] */ double TargetTime /* [out] */ __int64 *Result	Ожидание достижения заданного модельного времени, если проект запущен
PackStart	/* [in] */ __int64 PackId	Инициализировать пакет проектов
PackRun	/* [in] */ __int64 PackId	Запустить пакет проектов
PackStep	/* [in] */ __int64 PackId	Выполнить шаг расчета для пакета проектов
PackPause	/* [in] */ __int64 PackId	Поставить пакет проектов на паузу
PackStop	/* [in] */ __int64 PackId	Остановить пакет проектов

Имя функции	Список аргументов	Описание
RunToPack	/* [in] */ __int64 PackId /* [in] */ double TargetTime /* [out] */ __int64 *aResult	Запустить расчет пакета проектов до заданного времени
WaitForTimePack	/* [in] */ __int64 PackId /* [in] */ double TargetTime /* [out] */ __int64 *aResult	Ожидание достижения заданного модельного времени, если пакет проектов запущен
FormShow	/* [in] */ __int64 ProjectId	Показать форму проекта или пакета проектов
FormHide	/* [in] */ __int64 ProjectId	Скрыть форму проекта или пакета проектов
GetFormVisible	/* [in] */ __int64 ProjectId /* [out] */ long *Value	Получить флаг видимости формы проекта или пакета проектов
FormBringToFront	/* [in] */ __int64 ProjectId	Перенести форму проекта или пакета проектов на передний план
FormSendToBack	/* [in] */ __int64 ProjectId	Перенести форму проекта или пакета проектов на задний план
GetFormState	/* [in] */ __int64 ProjectId /* [out] */ long *Value	Получить состояние формы проекта или пакета проектов
SetFormState	/* [in] */ __int64 ProjectId /* [in] */ long Value	Установить состояние формы проекта или пакета проектов
GetFormStyle	/* [in] */ __int64 ProjectId /* [out] */ long *Value	Получить стиль формы проекта или пакета проектов
SetFormStyle	/* [in] */ __int64 ProjectId /* [in] */ long Value	Установить стиль формы для проекта или пакета проектов

Имя функции	Список аргументов	Описание
GetFormBorderStyle	/* [in] */ __int64 ProjectId /* [out] */ long *Value	Получить стиль границы формы проекта или пакета проектов
SetFormBorderStyle	/* [in] */ __int64 ProjectId /* [in] */ long Value	Установить стиль границы формы проекта или пакета проектов
GetFormHandle	/* [in] */ __int64 ProjectId /* [out] */ unsigned __int64 *FormHandle	Получить дескриптор формы проекта или пакета проектов
SetParentPrjHandle	/* [in] */ __int64 ProjectId /* [in] */ __int64 AHandle	Установить дескриптор родительской формы для формы проекта
SetPrjPosByPrjId	/* [in] */ __int64 SrcPrjId /* [in] */ __int64 DestPrjId	Установить положение окна проекта в соответствии с другим проектом
GetFormCoords	/* [in] */ __int64 ProjectId /* [out] */ long *cLeft /* [out] */ long *cTop /* [out] */ long *cRight /* [out] */ long *cBottom /* [out] */ double *Xcenter /* [out] */ double *Ycenter /* [out] */ double *cScale	Получить настройки отображения формы проекта
SetFormCoords	/* [in] */ __int64 ProjectId /* [in] */ long cLeft /* [in] */ long cTop /* [in] */ long cRight /* [in] */ long cBottom	Установить настройки отображения формы проекта

Имя функции	Список аргументов	Описание
	/* [in] */ double Xcenter /* [in] */ double Ycenter /* [in] */ double cScale /* [in] */ long Flags	
SetFormCaption	/* [in] */ __int64 ProjectId /* [in] */ BSTR ACaption	Установить подпись формы проекта
GetProjectTime	/* [in] */ __int64 ProjectId /* [out] */ double *PrjTime	Получить текущее модельное время проекта
SetProjectRealTimeDelay	/* [in] */ __int64 aProjectId /* [in] */ long aDelayFlag /* [in] */ double aDelayScale	Установить флаг синхронизации с реальным временем и коэффициент ускорения для пакета проектов
SetRealTimeDelayPack	/* [in] */ __int64 PackId /* [in] */ long aRTDelayFlag /* [in] */ double aRTDelayScale	Установить флаг синхронизации с реальным временем и коэффициент ускорения для пакета проектов
SetNoCloseFlag	/* [in] */ __int64 ProjectId /* [in] */ long Value	Установить флаг блокировки закрытия окна проекта
ClearProjectActions	/* [in] */ __int64 ProjectId	Очистить список действий проекта
RepaintEditor	/* [in] */ __int64 ProjectId	Полностью перерисовать окна проекта
GetOpenedFileName	/* [in] */ __int64 ProjectId /* [out] */ VARIANT *AFileName	Получить имя (путь) проекта
GetCmdStr	/* [in] */ __int64 ProjectId /* [out] */ VARIANT *CmdStr	Получить значение общей информационной строки проекта

Имя функции	Список аргументов	Описание
SetCmdStr	/* [in] */ __int64 ProjectId /* [in] */ BSTR CmdStr	Установить значение общей информационной строки проекта
SetProjectModified	/* [in] */ __int64 ProjectId /* [in] */ long AModified	Установить флаг – проект модифицирован
GetProjectStateFlag	/* [in] */ __int64 ProjectId /* [out] */ long *StateFlag	Получить флаг состояния проекта
GetEditorFlags	/* [in] */ __int64 ProjectId /* [out] */ long *aModifiedFlag /* [out] */ long *aSavedFlag /* [out] */ long *aCurPageModified /* [out] */ long *ReadOnly	Получить флаги состояния проекта
SetReadOnlyFlag	/* [in] */ __int64 ProjectId /* [in] */ long ReadOnlyFlag	Установить флаг открытия проекта только на чтение
ResetProjectEngines	/* [in] */ __int64 ProjectId	Сброс устройств менеджера данных
ShowAllBlocks	/* [in] */ __int64 ProjectId	Показать все блоки на текущей странице
SetSysProp	/* [in] */ __int64 ProjectId /* [in] */ long WindowLeft /* [in] */ long WindowTop /* [in] */ long WindowWidth /* [in] */ long WindowHeight /* [in] */ long fLoadRestart /* [in] */ BSTR LoadRestartFile /* [in] */ long fChangeRestartTime	Установить настройки формы и рестарта проекта

Имя функции	Список аргументов	Описание
	/* [in] */ double NewRestartTime /* [in] */ long fRestartSave /* [in] */ BSTR SaveRestartFile /* [in] */ long fSaveTimedRestart /* [in] */ double RestartStep /* [out] */ long *Result	
ReadProjectRestart	/* [in] */ __int64 ProjectId /* [in] */ BSTR FileName	Прочитать рестарт проекта из файла
WriteProjectRestart	/* [in] */ __int64 ProjectId /* [in] */ BSTR FileName	Записать рестарт проекта в файл
ReadRestartPoint	/* [in] */ __int64 ProjectId	Прочитать точку рестарта для проекта
WriteRestartPoint	/* [in] */ __int64 ProjectId	Записать точку рестарта для проекта
ReadPackRestart	/* [in] */ __int64 PackID /* [in] */ BSTR FileName	Загрузить рестарт пакета проектов
WritePackRestart	/* [in] */ __int64 PackID /* [in] */ BSTR FileName	Сохранить рестарт пакета проектов
SetProjectReadRestartFile	/* [in] */ __int64 ProjectId /* [in] */ BSTR FileName /* [in] */ long fLoadRst	Установить имя рестарта для чтения для проекта
SetProjectWriteRestartFile	/* [in] */ __int64 ProjectId /* [in] */ BSTR FileName /* [in] */ long fSaveRst	Установить имя рестарта для записи для проекта

Имя функции	Список аргументов	Описание
SetRestartPreserveFlag	/* [in] */ __int64 ProjectId /* [in] */ long fRestartPreserve	Установить флаг запрета сохранения рестарта для проекта
GetProjectRestartNames	/* [in] */ __int64 aPrjId /* [out] */ VARIANT *aReadRestartFile /* [out] */ VARIANT *aWriteRestartFile /* [out] */ long *aReadRestartFlag /* [out] */ long *aWriteRestartFlag /* [out] */ double *aNewRestartTime /* [out] */ long *aSetNewTimeFlag	Получить настройки рестарта для проекта
GetProjectDB	/* [in] */ __int64 ProjectId /* [out] */ VARIANT *DBPlugin /* [out] */ VARIANT *DBName	Получить сведения о базе данных, подключенной к проекту
SetDBOverride	/* [in] */ BSTR aOverrideDBPlugin /* [in] */ BSTR aOverrideDBName	Установить переопределение загрузки базы данных
ReloadProjectDB	/* [in] */ __int64 ProjectId /* [in] */ BSTR aDBPluginName /* [in] */ BSTR aDBName	Установить имена базы данных и модуля базы данных проекта и перезагрузить
ExportDBToXML	/* [in] */ __int64 ProjectId /* [in] */ BSTR DBFileName /* [out] */ long *Result	Экспортировать базу данных проекта в XML
FindProjectData	/* [in] */ BSTR DataName /* [in] */ __int64 ProjectId /* [in] */ long AccesForWrite	Поиск элемента данных в проекте по его имени

Имя функции	Список аргументов	Описание
	/* [out] */ struct TDataDescriptor *DataDesc	
FindSignalData	/* [in] */ BSTR SignalName /* [in] */ __int64 ProjectId /* [out] */ TDataDescriptor *DataDesc	Поиск элемента данных в сигналах проекта и базе данных по его имени
FindPackSignal	/* [in] */ __int64 PackId /* [in] */ BSTR SignalName /* [out] */ TDataDescriptor *DataDesc	Поиск элемента данных в сигналах пакетах проекта и базе данных по его имени
GetProjectSignalList	/* [in] */ __int64 ProjectId /* [out] */ __int64 *ListId	Получить список сигналов проекта
GetPackSignalList	/* [in] */ __int64 PackId /* [out] */ __int64 *ListId	Получить список сигналов пакета проектов
GetListCount	/* [in] */ __int64 ListId /* [out] */ __int64 *Count	Получить количество элементов в списке
GetDataInfoFromList	/* [in] */ __int64 ListId /* [in] */ long ElementNumber /* [out] */ VARIANT *DataName /* [out] */ VARIANT *DataCaption /* [out] */ TDataDescriptor *DataDesc	Получить элемент данных из списка по индексу
FindDataInListByName	/* [in] */ __int64 ListId /* [in] */ BSTR Name /* [out] */ long *ElementNumber	Получить индекс элемента в списке по имени элемента данных
ReadAsFloat	/* [in] */ struct TDataDescriptor DataDesc /* [out] */ double *Result	Чтение элемента данных вещественного типа по дескриптору

Имя функции	Список аргументов	Описание
ReadAsInteger	/* [in] */ struct TDataDescriptor DataDesc /* [out] */ __int64 *Result	Чтение элемента данных целого типа по дескриптору
ReadAsString	/* [in] */ struct TDataDescriptor DataDescriptor /* [out] */ VARIANT *Result	Чтение элемента данных строкового типа по дескриптору
GetArrayCount	/* [in] */ struct TDataDescriptor DataDesc /* [out] */ long *Result	Получить размерность массива по дескриптору
GetExtArrayElement	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ long Index /* [out] */ double *Result	Получить элемент вещественного массива по дескриптору
GetIntArrayElement	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ long Index /* [out] */ __int64 *Result	Получить элемент целочисленного массива по дескриптору
WriteAsFloat	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ double Source	Запись элемента данных вещественного типа по дескриптору
WriteAsInteger	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ __int64 Source	Запись элемента данных целого типа по дескриптору
WriteAsString	/* [in] */ struct TDataDescriptor DataDescriptor /* [in] */ BSTR Source	Запись элемента данных строкового типа по дескриптору
WriteAsFont	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ BSTR FontName /* [in] */ long FontSize /* [in] */ byte FontStyle	Запись элемента данных типа шрифт по дескриптору
SetArrayCount	/* [in] */ struct TDataDescriptor DataDesc	Установить размерность массива по дескриптору

Имя функции	Список аргументов	Описание
	/* [in] */ long Count	
SetExtArrayElement	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ long Index /* [in] */ double Source	Установить элемент вещественного массива по дескриптору
SetIntArrayElement	/* [in] */ struct TDataDescriptor DataDesc /* [in] */ long Index /* [in] */ __int64 Source	Установить элемент целочисленного массива по дескриптору
OpenExchangeFile	/* [in] */ enum TExchangeMethod ExchangeMethod /* [in] */ BSTR FileName /* [out] */ __int64 *DataObjId	Создание стандартных устройств обмена данными
CloseExchangeFile	/* [in] */ __int64 *EngineId	Закрытие устройства обмена данными
Read	/* [in] */ THandleArray Handles /* [in] */ __int64 EngineId	Запись массива данных в устройство обмена согласно полученным дескрипторам
Write	/* [in] */ THandleArray Handles /* [in] */ __int64 EngineId	Запись массива данных из устройства обмена согласно полученным дескрипторам
ClearReadList		Очистка внутреннего списка дескрипторов на чтение
ClearWriteList		Очистка внутреннего списка дескрипторов на запись
AddToReadList	/* [in] */ struct TDataDescriptor DataDesc	Добавление элемента данных во внутренний список дескрипторов на чтение
AddToWriteList	/* [in] */ struct TDataDescriptor DataDesc	Добавление элемента данных во внутренний список дескрипторов на запись
ReadList	/* [in] */ __int64 EngineId	Запись массива данных в устройство обмена согласно внутреннему списку дескрипторов

Имя функции	Список аргументов	Описание
WriteList	/* [in] */ __int64 EngineId	Запись массива данных из устройства обмена согласно внутреннему списку дескрипторов
GetLayerName	/* [in] */ __int64 ProjectId /* [in] */ long LayerNumber /* [out] */ VARIANT *LayerName	Получить имя расчетного слоя проекта по индексу
SetLayerProp	/* [in] */ __int64 ProjectId /* [in] */ long LayerNo /* [in] */ BSTR PropName /* [in] */ BSTR StrValue /* [out] */ __int64 *Result	Установить значение свойства расчетного слоя проекта
SendDataToLayer	/* [in] */ __int64 ProjectId /* [in] */ long LayerID /* [in] */ long Command /* [in] */ BSTR InputData /* [out] */ __int64 *Response /* [out] */ VARIANT *OutData	Послать произвольные данные в расчетный слой
SendPluginCommand	/* [in] */ BSTR PluginName /* [in] */ long CommandId /* [in] */ BSTR CommandStr /* [in] */ __int64 ObjId /* [out] */ VARIANT *ResultStr /* [out] */ __int64 *ResultPtr	Послать произвольную команду загруженному плагину

Имя функции	Список аргументов	Описание
GetMainPage	/* [in] */ __int64 ProjectId /* [out] */ __int64 *PageId	Получить уникальный идентификатор (ссылку) главной страницы проекта
GetCurentPage	/* [in] */ __int64 ProjectId /* [out] */ __int64 *PageId	Получить уникальный идентификатор (ссылку) текущей страницы
SetCurrentPage	/* [in] */ __int64 ProjectId /* [in] */ __int64 PageId /* [out] */ __int64 *Result	Установить текущую страницу проекта
PageUp	/* [in] */ __int64 PageId /* [out] */ __int64 *UpPageId	Подняться на уровень выше от заданной страницы
SetPageCoords	/* [in] */ __int64 PageId /* [in] */ double X /* [in] */ double Y /* [in] */ double Scale /* [out] */ long *Result	Установить положение отображения для страницы
SetPageWindow	/* [in] */ __int64 PageId /* [in] */ long Left /* [in] */ long Top /* [in] */ long Width /* [in] */ long Height /* [out] */ long *Result	Установить размеры окна индивидуально для страницы
SetPageScript	/* [in] */ __int64 ProjectId /* [in] */ BSTR Script /* [in] */ long CompileNow	Установить скрипт для текущей страницы

Имя функции	Список аргументов	Описание
	/* [out] */ long *Result	
GetPageObjectCount	/* [in] */ __int64 ProjectId /* [out] */ long *ObjCount	Получить количество элементов на текущей странице проекта
GetPageBlockId	/* [in] */ __int64 ProjectId /* [in] */ long BlockIndex /* [out] */ __int64 *BlockId	Получить уникальный идентификатор (ссылку) блока по индексу
GetSubmodelPage	/* [in] */ __int64 BlockId /* [out] */ __int64 *PageId	Получить уникальный идентификатор (ссылку) страницы для блока-субмодели
CreateBlock	/* [in] */ __int64 ProjectId /* [in] */ long LayerNo /* [in] */ __int64 ParentBlock /* [in] */ BSTR LibRecordName /* [out] */ __int64 *BlockId	Создать новый блок на текущей странице проекта
InitBlock	/* [in] */ __int64 BlockId	Инициализировать блок
BlockAfterEdit	/* [in] */ __int64 BlockId	Вызов секции после редактирования блока
LoadSubmodel	/* [in] */ __int64 BlockId /* [in] */ BSTR FileName	Загрузить субмодель из файла
AssignSubmodel	/* [in] */ __int64 ProjectId /* [in] */ __int64 BlockId /* [in] */ BSTR FileName	Загрузить и связать субмодель с файлом
SetBlockPosition	/* [in] */ __int64 BlockId /* [in] */ double Left /* [in] */ double Top	Установить положение блока

Имя функции	Список аргументов	Описание
	/* [in] */ double Width /* [in] */ double Height /* [in] */ double Angle /* [out] */ long *Result	
GetBlockPropAsString	/* [in] */ __int64 BlockId /* [in] */ BSTR PropName /* [out] */ VARIANT *ValueAsString	Получить значение свойства блока как строку
GetBlockPluginName	/* [in] */ __int64 BlockId /* [out] */ VARIANT *PluginClassName	Получить имя класса модуля расширения (плагина) блока
GetBlockCalcTemplate	/* [in] */ __int64 BlockId /* [out] */ VARIANT *CalcTemplateStr	Получить конфигурацию расчетного шаблона блока
SetBlockProp	/* [in] */ __int64 BlockId /* [in] */ BSTR PropName /* [in] */ BSTR StrValue /* [out] */ __int64 *Result	Установить значение настраиваемого свойства блока
SetGraphBlockProp	/* [in] */ __int64 BlockId /* [in] */ BSTR PropName /* [in] */ BSTR StrValue /* [out] */ __int64 *Result	Установить значение системного свойства блока
GetPropHandle	/* [in] */ __int64 BlockId /* [in] */ BSTR PropName /* [out] */ __int64 *PropHandle	Получить ссылку на элемент данных по имени настраиваемого свойства

Имя функции	Список аргументов	Описание
GetGraphPropHandle	/* [in] */ __int64 BlockId /* [in] */ BSTR PropName /* [out] */ __int64 *PropHandle	Получить ссылку на элемент данных по имени системного свойства
ExecutePropScript	/* [in] */ __int64 BlockId /* [in] */ __int64 DataId /* [out] */ long *Result	Выполнить скрипт для свойства при его изменении (например, изменить число портов в зависимости от параметра)
SetFontData	/* [in] */ __int64 PropHandle /* [in] */ BSTR FontName /* [in] */ long Height /* [in] */ long Color /* [in] */ byte Style	Установить настройки шрифта по ссылке на элемент данных
CreatePrimitiv	/* [in] */ __int64 ProjectId /* [in] */ long LayerNo /* [in] */ __int64 ParentBlock /* [in] */ long PrimitivId /* [out] */ __int64 *BlockId	Создать графический примитив на текущей странице проекта
CreateWire	/* [in] */ __int64 ProjectId /* [in] */ long LayerNo /* [in] */ long WireType /* [in] */ __int64 ParentWire /* [in] */ long ParentPointNo /* [in] */ __int64 StartPort /* [in] */ __int64 EndPort	Создать линию связи

Имя функции	Список аргументов	Описание
	/* [in] */ long PointCount /* [out] */ __int64 *WireId	
NormalizeWire	/* [in] */ __int64 WireId	Выровнять линию связи
SetWirePoint	/* [in] */ __int64 WireId /* [in] */ long PointNo /* [in] */ double X /* [in] */ double Y /* [out] */ long *Result	Установить положение опорной точки линии связи
GetPointCount	/* [in] */ __int64 BlockId /* [out] */ long *Count	Получить количество точек для блока
AddObjectPoint	/* [in] */ __int64 BlockId /* [in] */ double X /* [in] */ double Y	Добавить точку для блока
InsertObjectPoint	/* [in] */ __int64 BlockId /* [in] */ long Index /* [in] */ double X /* [in] */ double Y	Вставить точку для блока
DeleteObjectPoint	/* [in] */ __int64 BlockId /* [in] */ long Index /* [in] */ long Count	Удалить точку для блока
GetBlockEngine	/* [in] */ __int64 BlockId /* [out] */ __int64 *EngineId	Получить ссылку устройства, ассоциированного с блоком
SetGraphicView	/* [in] */ __int64 EngineId	Установить общие параметры графика

Имя функции	Список аргументов	Описание
	/* [in] */ unsigned long Color /* [in] */ long Left /* [in] */ long Top /* [in] */ long Width /* [in] */ long Height /* [in] */ long FormMode /* [in] */ BSTR PlotCaption /* [in] */ BSTR MainLabel /* [in] */ BSTR XLabel /* [in] */ BSTR YLabel /* [in] */ long Visible	
GetPortCount	/* [in] */ __int64 BlockId /* [out] */ long *Count	Получить количество портов блока
SetPortCount	/* [in] */ __int64 BlockId /* [in] */ long Count /* [in] */ long DefaultMode /* [in] */ long DefaultType /* [in] */ long DefaultSide /* [out] */ long *Result	Изменить количество портов у блока
SetCondPortCount	/* [in] */ __int64 BlockId /* [in] */ long Count /* [in] */ long Mode /* [in] */ long AType	Изменить количество портов, удовлетворяющих заданным условиям

Имя функции	Список аргументов	Описание
	/* [in] */ long DefaultSide /* [in] */ BSTR AName /* [in] */ long APortVariant /* [out] */ long *Result	
GetBlockPort	/* [in] */ __int64 BlockId /* [in] */ long Index /* [out] */ __int64 *PortId	Получить уникальный идентификатор (ссылку) порта по номеру
GetInPort	/* [in] */ __int64 BlockId /* [in] */ long InNo /* [out] */ __int64 *PortId	Получить идентификатор входа блока по его номеру
GetOutPort	/* [in] */ __int64 BlockId /* [in] */ long OutNo /* [out] */ __int64 *PortId	Получить идентификатор выхода блока по его номеру
GetPortInfo	/* [in] */ __int64 PortId /* [out] */ VARIANT *Name /* [out] */ long *Side /* [out] */ long *Mode /* [out] */ long *TypeId /* [out] */ long *ItemId /* [out] */ long *IsInverse /* [out] */ long *IsCenter /* [out] */ long *IsInvisible /* [out] */ double *X	Получить информацию о порте

Имя функции	Список аргументов	Описание
	/* [out] */ double *Y /* [out] */ double *XGlobal /* [out] */ double *YGlobal	
SetPortName	/* [in] */ __int64 PortId /* [in] */ BSTR PortName /* [out] */ long *Result	Установить имя для порта
SetPortSide	/* [in] */ __int64 PortId /* [in] */ long Side /* [out] */ long *Result	Установить положение порта блока (слева, справа, снизу, сверху)
SetPortInverse	/* [in] */ __int64 PortId /* [in] */ long Inverse /* [out] */ long *Result	Установить вид для порта
SetPortItemId	/* [in] */ __int64 PortId /* [in] */ long ItemId /* [out] */ long *Result	Установить вариант для порта
SetPortMode	/* [in] */ __int64 PortId /* [in] */ long Mode /* [out] */ long *Result	Установить режим для порта
SetPortLineTypeId	/* [in] */ __int64 PortId /* [in] */ long LineTypeId /* [out] */ long *Result	Установить тип линии связи для порта
SetPortInvisible	/* [in] */ __int64 PortId /* [in] */ long Invisible	Установить флаг видимости для порта

Имя функции	Список аргументов	Описание
	/* [out] */ long *Result	

Далее на примере простейшего консольного приложения на языке C++ продемонстрирован принцип взаимодействия с API SimInTech, используя технологию COM.

Заголовочные файлы *mmain.hpp* и *mmain_i.c*, сгенерированные самостоятельно или взятые в директории *$(SIT)\source\exe*, добавляются в состав проекта в Microsoft Visual Studio. Используемая в основе механизма взаимодействия, функция *CoCreateInstance* получает идентификатор CLSID *(CLSID_MVTU_Server* описан в файле *mmain_i.c)*, создает соответствующий компонент и возвращает требуемый указатель на интерфейс (*IMVTU_Server* MVTU)*. *CoCreateInstance* в процессе выполнения вызывает *CoGetClassObject* и получает указатель на интерфейс *IClassFactory* фабрики класса. Затем с помощью полученного указателя *CoCreateInstance* вызывает *IClassFactory::CreateInstance*, результатом чего и является создание нового компонента. Таким образом, *CoCreateInstance* запускает приложение с использованием *GUID*, возвращая указатель на интерфейс *IMVTU_Server* MVTU*.

Разработанное консольное приложение дает возможность пользователю выполнять следующие действия с загруженным проектом или пакетом проектов, используя API SimInTech:

- транслировать в консоль лог программы;
- инициализировать модель;
- выполнить шаг расчета модели;
- запустить расчет модели;
- установить паузу расчета модели;
- остановить расчет модели;
- загрузить рестарт для модели;
- сохранить рестарт для модели;
- задавать настройки синхронизации с реальным временем;
- экспортировать базу данных модели в XML;
- получить путь к модели;

- включить режим работы программы в скрытом виде без пользовательского интерфейса (*silentmode*);
- выключить режим работы программы в скрытом виде без пользовательского интерфейса (*silentmode*);
- получить текущее модельное время;
- получить значение сигнала (базы данных) модели;
- установить значение сигнала (базы данных) модели.

Ниже приведен код реализации основного модуля консольного приложения на языке C++ с использованием среды разработки Microsoft Visual Studio 2022 (исходные коды данного приложения продублированы в директории *$(SIT)\source\API Sample*):

```
#include <iostream>
#include <string>
#include <thread>
#include <windows.h>
#include <comutil.h>
#include <stdio.h>
#include "mmain.hpp"
#pragma warning(disable : 4996) //_CRT_SECURE_NO_WARNINGS
#pragma comment(lib, "comsuppw.lib")

/* wchar_t to char */
char* wtoc(const wchar_t* w, size_t max)
{
	char* c = new char[max];
	wcstombs(c, w, max);
	return c;
}

/* char to wchar_t */
wchar_t* ctow(const char* c, size_t max)
{
	wchar_t* w = new wchar_t[max];
	mbstowcs(w, c, max);
	return w;
}

/* get filename's exp */
size_t find_ext_idx(const char* fileName)
{
	size_t len = strlen(fileName);
	size_t idx = len - 1;
	for (size_t i = 0; *(fileName + i); i++) {
```

```
			if (*(fileName + i) == '.') {
				idx = i;
			}
			else if (*(fileName + i) == '/' || *(fileName + i) == '\\') {
				idx = len - 1;
			}
		}
		return idx + 1;
}

std::string get_file_ext(const char* fileName)
{
		return std::string(fileName).substr(find_ext_idx(fileName));
}

//Запустить приложение через COM
HRESULT LaunchApp(
		_In_ REFCLSID rclsid,
		_In_opt_ LPUNKNOWN pUnkOuter,
		_In_ DWORD dwClsContext,
		_In_ REFIID riid,
		_COM_Outptr_ _At_(*ppv, _Post_readable_size_(_Inexpressible_(varies))) LPVOID  FAR* ppv)

{
		CoInitialize(NULL);
		return CoCreateInstance(
			rclsid,
			pUnkOuter,
			dwClsContext,
			riid,
			ppv);
}

//Загрузить модель
bool LoadModel(
		std::string aModelPath,
		IMVTU_Server* aMVTU,
		_int64* aProjectID,
		_int64* aPackID,
		bool* aIsPack)
{
		*aProjectID = 0;
		*aPackID = 0;
		*aIsPack = false;
		if (get_file_ext(aModelPath.data()) == "pak") {
			*aIsPack = true;
```

```
            BSTR temp = _com_util::ConvertStringToBSTR(aModelPath.c_str());
            aMVTU->OpenPack(temp, aPackID);
            SysFreeString(temp);
            aMVTU->PackGetProjectIdByIndex(*aPackID, 0, aProjectID);
      }
      else
      if ((get_file_ext(aModelPath.data()) == "prt") || (get_file_ext(aModelPath.data()) == "xprt")) {
            BSTR temp = _com_util::ConvertStringToBSTR(aModelPath.c_str());
            aMVTU->OpenProject(temp, aProjectID);
            SysFreeString(temp);
      }
      return *aProjectID != 0;
}

//Инициализировать модель
void InitModel(
      IMVTU_Server* aMVTU,
      _int64 aProjectID,
      _int64 aPackID,
      bool aIsPack)
{
      if (aIsPack) aMVTU->PackStart(aPackID);
      else         aMVTU->ProjectStart(aProjectID);
}

//Выполнить шаг расчета модели
void StepModel(
      IMVTU_Server* aMVTU,
      _int64 aProjectID,
      _int64 aPackID,
      bool aIsPack)
{
      if (aIsPack) aMVTU->PackStep(aPackID);
      else         aMVTU->ProjectStep(aProjectID);
}

//Запустить расчет модели
void RunModel(
      IMVTU_Server* aMVTU,
      _int64 aProjectID,
      _int64 aPackID,
      bool aIsPack)
{
      if (aIsPack) aMVTU->PackRun(aPackID);
      else         aMVTU->ProjectRun(aProjectID);
}
```

```
//Выставить паузу расчета модели
void PauseModel(
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack)
{
	if (aIsPack) aMVTU->PackPause(aPackID);
	else         aMVTU->ProjectPause(aProjectID);
}

//Остановить расчет модели
void StopModel(
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack)
{
	if (aIsPack) aMVTU->PackStop(aPackID);
	else         aMVTU->ProjectStop(aProjectID);
}

//Закрыть модель
void CloseModel(
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack)
{
	StopModel(aMVTU, aProjectID, aPackID, aIsPack);
	if (aIsPack) aMVTU->ClosePack(aPackID);
	else         aMVTU->CloseProject(aProjectID);
}

//Загрузить рестарт для модели
void ReadRestart(
	std::string aBaseName,
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack)
{
	BSTR temp = _com_util::ConvertStringToBSTR(aBaseName.c_str());
	if (aIsPack) aMVTU->ReadPackRestart(aPackID, temp);
	else         aMVTU->ReadProjectRestart(aProjectID, temp);
```

```
	SysFreeString(temp);
}

//Сохранить рестарт для модели
void WriteRestart(
	std::string aBaseName,
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack)
{
	BSTR temp = _com_util::ConvertStringToBSTR(aBaseName.c_str());
	if (aIsPack) aMVTU->WritePackRestart(aPackID, temp);
	else         aMVTU->WriteProjectRestart(aProjectID, temp);
	SysFreeString(temp);
}

//Экспортировать базу данных модели в XML
bool ExportDBAsXML(
	std::string aFileName,
	IMVTU_Server* aMVTU,
	_int64 aProjectID)
{
	long R = 0;
	if (get_file_ext(aFileName.data()) == "xml") {
		BSTR temp = _com_util::ConvertStringToBSTR(aFileName.c_str());
		aMVTU->ExportDBToXML(aProjectID, temp, &R);
		SysFreeString(temp);
	}
	return R == 1;
}

//Установить синхронизацию с реальным временем
void SetKAcceleration(
	IMVTU_Server* aMVTU,
	_int64 aProjectID,
	_int64 aPackID,
	bool aIsPack,
	bool aSyncWithRealTime,
	double aKAcc)
{
	if (aIsPack) aMVTU->SetRealTimeDelayPack(aPackID, aSyncWithRealTime, aKAcc);
	else         aMVTU->SetProjectRealTimeDelay(aProjectID, aSyncWithRealTime, aKAcc);
}

//Закрыть процесс по PID
```

```
void CloseProcess(unsigned long aPID)
{
	HANDLE H = NULL;
	H = OpenProcess(SYNCHRONIZE | PROCESS_TERMINATE, true, aPID);
	if (H != NULL) {
		TerminateProcess(H, 0);
		CloseHandle(H);
	}
}

//Процедура для вывода в консоль отладочной информации из программы
void DebugProc(std::string aPipeName)
{
	aPipeName = "\\\\.\\pipe\\" + aPipeName;

	const long symb_count = 4096;
	HANDLE hPipe;
	BSTR temp = _com_util::ConvertStringToBSTR(aPipeName.c_str());
	hPipe = CreateNamedPipe(
		temp,
		PIPE_ACCESS_INBOUND,
		PIPE_TYPE_BYTE | PIPE_READMODE_BYTE | PIPE_WAIT,
		10,
		0,
		symb_count * sizeof(wchar_t),
		NMPWAIT_USE_DEFAULT_WAIT,
		NULL);
	SysFreeString(temp);

	if (hPipe != INVALID_HANDLE_VALUE)
	{
		wchar_t buffer[symb_count];
		DWORD dwRead;
		if (ConnectNamedPipe(hPipe, NULL))
			while (true)
			{
				if (ReadFile(hPipe, buffer, sizeof(buffer), &dwRead, NULL))
				{
					if (dwRead < sizeof(buffer))
						buffer[dwRead / sizeof(wchar_t)] = '\0';

					printf("\n___LOG_THREAD:_____________________________________\n");
					printf("%s", wtoc(buffer, symb_count));
					printf("_____________________________________________________\n\n");
				}
				else
```

```
                                    break;
                        }
        }

        if (hPipe != INVALID_HANDLE_VALUE)
                DisconnectNamedPipe(hPipe);
}

int main()
{

        //Настройки консоли
        SetConsoleCP(1251);
        SetConsoleOutputCP(1251);
        _wsetlocale(LC_ALL, L"Russian");
        setlocale(LC_ALL, "Russian");

        printf("Type enter to launch SimInTech\n");
        std::cin.get();

        //Запустить приложение через COM
        IMVTU_Server* MVTU = NULL;
        if (FAILED(LaunchApp(CLSID_MVTU_Server, NULL, CLSCTX_LOCAL_SERVER, IID_IMVTU_Server, (void**)&MVTU)))
                printf("CoCreateInstance failed\n");

        if (MVTU != NULL) {

                //Получить PID
                unsigned long App_PID = 0;
                MVTU->GetProcessID(&App_PID);
                //Нстройки приложения
                MVTU->SetNoCloseAppFlag(1);
                MVTU->SetShutdownOnLastRelease(1);

                printf("Type model abs path (*.prt, *.xprt, *.pak):\n");
                std::string Path;
                std::getline(std::cin, Path);

                __int64 Project_ID = 0;
                __int64 Pack_ID = 0;
                bool IsPack = false;
                //Загрузить модель
                if (!LoadModel(Path, MVTU, &Project_ID, &Pack_ID, &IsPack)) {
                        printf("Incorrect file\n");
                }
                else {
```

```cpp
bool In_Loop = true;
int CMD;
char SName[20];
double SDVal;
__int64 SIVal;
char SSVal[50];
VARIANT V;
long Count;
bool SyncWithRealTime;
double KAcceleration;
struct TDataDescriptor SDataDesc;
BSTR temp;

//Настройки для перехвата лога программы
std::string PipeName = "SITDebugPipe";
MVTU->SetJournalSavePeriod(1000);
temp = _com_util::ConvertStringToBSTR(PipeName.c_str());
MVTU->SetPipeName(temp);
SysFreeString(temp);
//Поток для вывода в консоль лога программы
std::thread th1(DebugProc, PipeName);

while (In_Loop) {
	printf("Type CMD (0-finish, 1-init, 2-step, 3-run, 4-pause, 5-stop, 6-readrst, 7-writerst, "
		"8-setsync, 9-exportdbtoxml, 10-getpath, 11-silentmodeon, 12-silentmodeoff, 13-gettime, "
		"14-getval, 15-setval):\n");
	std::cin >> CMD;

	switch (CMD) {
	//Выйти из цикла
	case 0:
		In_Loop = false;
		CloseModel(MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Инициализировать модель
	case 1:
		InitModel(MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Выполнить шаг расчета модели
	case 2:
		StepModel(MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Запустить расчет модели
	case 3:
		RunModel(MVTU, Project_ID, Pack_ID, IsPack);
```

```
		break;
	//Выставить паузу расчета модели
	case 4:
		PauseModel(MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Остановить расчет модели
	case 5:
		StopModel(MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Загрузить рестарт для модели
	case 6:
		printf("Type base rst name:\n");
		std::cin.get();
		std::getline(std::cin, Path);
		ReadRestart(Path, MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Сохранить рестарт для модели
	case 7:
		printf("Type base rst name:\n");
		std::cin.get();
		std::getline(std::cin, Path);
		WriteRestart(Path, MVTU, Project_ID, Pack_ID, IsPack);
		break;
	//Установить настройки синхронизации с реальным временем
	case 8:
		printf("Type sync with real time flag:\n");
		std::cin >> SyncWithRealTime;
		KAcceleration = 1;
		if (SyncWithRealTime) {
			printf("Type acceleration k:\n");
			std::cin >> KAcceleration;
		}
		SetKAcceleration(MVTU, Project_ID, Pack_ID, IsPack, SyncWithRealTime, KAcceleration);
		break;
	//Экспортировать базу данных модели в XML
	case 9:
		printf("Type db export abs path (*.xml):\n");
		std::cin.get();
		std::getline(std::cin, Path);
		if (!ExportDBAsXML(Path, MVTU, Project_ID))
			printf("Incorrect file\n");
		break;
	//Получить путь к модели
	case 10:
		MVTU->GetOpenedFileName(Project_ID, &V);
		printf(wtoc(V.bstrVal, 256));
```

```
		printf("\n");
		break;
	//Включить silentmode
	case 11:
		MVTU->SetSilentMode(true);
		break;
	//Выключить silentmode
	case 12:
		MVTU->SetSilentMode(false);
		break;
	//Получить текущее модельное время
	case 13:
		MVTU->GetProjectTime(Project_ID, &SDVal);
		printf("%.5lf\n", SDVal);
		break;
	//Получить значение сигнала модели
	case 14:
		printf("Type signal name (bool, int, double, intarray, array, string):\n");
		std::cin >> SName;
		temp = _com_util::ConvertStringToBSTR(SName);
		MVTU->FindSignalData(temp, Project_ID, &SDataDesc);
		SysFreeString(temp);
		if (SDataDesc.DataId != 0) {
			switch (SDataDesc.DataType) {
			//double
			case(0):
				MVTU->ReadAsFloat(SDataDesc, &SDVal);
				printf("%.5lf\n", SDVal);
				break;
			//integer
			case(1):
				MVTU->ReadAsInteger(SDataDesc, &SIVal);
				printf("%i\n", int(SIVal));
				break;
			//boolean
			case(2):
				MVTU->ReadAsInteger(SDataDesc, &SIVal);
				printf("%i\n", int(SIVal));
				break;
			//string
			case(4):
				MVTU->ReadAsString(SDataDesc, &V);
				printf(wtoc(V.bstrVal, 256));
				printf("\n");
				break;
			//array
```

```
                case(5):
                        MVTU->GetArrayCount(SDataDesc, &Count);
                        printf("%i\n", Count);
                        for (int i = 0; i < Count; i++) {
                                MVTU->GetExtArrayElement(SDataDesc, i, &SDVal);
                                printf("%.5lf ", SDVal);
                        }
                        printf("\n");
                        break;
                //intarray
                case(12):
                        MVTU->GetArrayCount(SDataDesc, &Count);
                        printf("%i\n", Count);
                        for (int i = 0; i < Count; i++) {
                                MVTU->GetIntArrayElement(SDataDesc, i, &SIVal);
                                printf("%i ", int(SIVal));
                        }
                        printf("\n");
                        break;
                default:
                        printf("Incorrect data type\n");
                }
        }
        else
                printf("Signal is not found\n");
        break;
//Установить значение сигнала модели
case 15:
        printf("Type signal name (bool, int, double, intarray, array, string):\n");
        std::cin >> SName;
        temp = _com_util::ConvertStringToBSTR(SName);
        MVTU->FindSignalData(temp, Project_ID, &SDataDesc);
        SysFreeString(temp);
        if (SDataDesc.DataId != 0) {
                switch (SDataDesc.DataType) {
                //double
                case(0):
                        printf("Type signal value (double):\n");
                        std::cin >> SDVal;
                        MVTU->WriteAsFloat(SDataDesc, SDVal);
                        break;
                //integer
                case(1):
                        printf("Type signal value (int):\n");
                        std::cin >> SIVal;
                        MVTU->WriteAsInteger(SDataDesc, SIVal);
```

```
                        break;
                    //boolean
                    case(2):
                        printf("Type signal value (bool):\n");
                        std::cin >> SIVal;
                        MVTU->WriteAsInteger(SDataDesc, SIVal);
                        break;
                    //string
                    case(4):
                        printf("Type signal value (string):\n");
                        std::cin >> SSVal;
                        temp = _com_util::ConvertStringToBSTR(SSVal);
                        MVTU->WriteAsString(SDataDesc, temp);
                        SysFreeString(temp);
                        break;
                    //array
                    case(5):
                        printf("Type array dim:\n");
                        std::cin >> Count;
                        MVTU->SetArrayCount(SDataDesc, Count);
                        for (int i = 0; i < Count; i++) {
                            printf("Type '%i' element value:\n", i);
                            std::cin >> SDVal;
                            MVTU->SetExtArrayElement(SDataDesc, i, SDVal);
                        }
                        break;
                    //intarray
                    case(12):
                        printf("Type intarray dim:\n");
                        std::cin >> Count;
                        MVTU->SetArrayCount(SDataDesc, Count);
                        for (int i = 0; i < Count; i++) {
                            printf("Type '%i' element value:\n", i);
                            std::cin >> SIVal;
                            MVTU->SetIntArrayElement(SDataDesc, i, SIVal);
                        }
                        break;
                    default:
                        printf("Incorrect data type\n");
                    }
                }
                else
                    printf("Signal is not found\n");
                break;
            default:
                printf("Incorrect CMD\n");
```

```
                        }
                }

                //Отключение потока вывода лога в консоль
                th1.detach();
                //Восстановление дефолтных настроек записи лога
                MVTU->SetJournalSavePeriod(60000);
                temp = _com_util::ConvertStringToBSTR("");
                MVTU->SetPipeName(temp);
                SysFreeString(temp);

        }

        //Отключиться от COM объекта
        MVTU->Release();
        //Дополнительно закрыть процесс по PID, т.к. MVTU->Release() не всегда закрывает приложение
        CloseProcess(App_PID);

    }

    return 0;

}
```