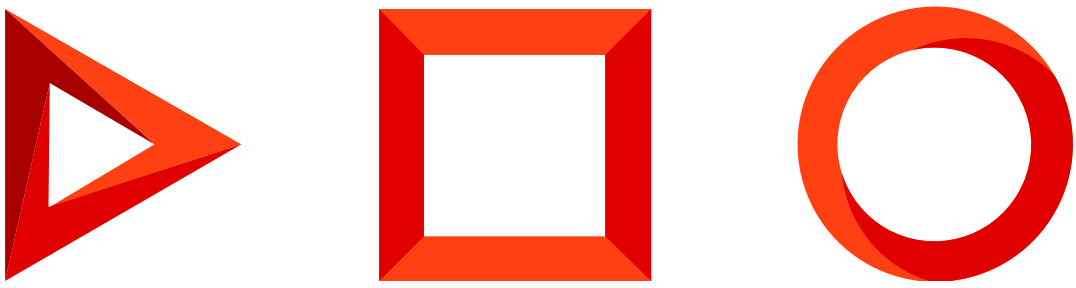
Creatio Academy

Интеграция с телефонией

Версия 8.0



Эта документация предоставляется с ограничениями на использование и защищена законами об интеллектуальной собственности. За исключением случаев, прямо разрешенных в вашем лицензионном соглашении или разрешенных законом, вы не можете использовать, копировать, воспроизводить, переводить, транслировать, изменять, лицензировать, передавать, распространять, демонстрировать, выполнять, публиковать или отображать любую часть в любой форме или посредством любые значения. Обратный инжиниринг, дизассемблирование или декомпиляция этой документации, если это не требуется по закону для взаимодействия, запрещены.

Информация, содержащаяся в данном документе, может быть изменена без предварительного уведомления и не может гарантировать отсутствие ошибок. Если вы обнаружите какие-либо ошибки, сообщите нам о них в письменной форме.

Содержание

Общие принципы работы интеграции с телефонией

Сложный

Creatio предоставляет возможность интеграции с рядом автоматических телефонных станций (АТС, Private Branch Exchange, PBX) для управления звонками непосредственно из интерфейса системы. Функциональность телефонии представлена в интерфейсе в виде CTI-панели (Computer Telephony Integration), а также в виде раздела [*Звонки*]. Стандартные возможности CTI-панели:

- отображение пользователю входящего звонка с функцией поиска контакта/контрагента по номеру звонящего;
- выполнение звонка из системы в один клик;
- управление звонком в Creatio (ответить, поставить/снять с удержания, завершить, перевести);
- отображение истории звонков для удобного управления связями звонков или возможности перезвонить.

Все звонки, которые были выполнены или приняты с использованием интеграции, сохраняются в разделе [*Звонки*]. В разделе можно посмотреть временные характеристики звонка (дата начала, дата завершения, длительность разговора), а также с чем связан звонок в системе.

Для работы со звонками в приложении необходимо настроить интеграцию с телефонией.

В зависимости от АТС, с которой выполняется интеграция, и особенностей предоставляемого API (Application Program Interface) используются разные архитектурные механизмы, которые описаны ниже. Также в зависимости от выбранного API может отличатся набор возможностей. Например, функция прослушивания звонков доступна не для всех АТС, а возможность использования Web-телефона доступна только при интеграции с Webitel. Вне зависимости от выбранного механизма интеграции, интерфейс CTI-панели остается одинаковым для всех пользователей.

Способы интеграции с системой обмена сообщениями в Creatio

Способы интеграции можно разделить на два типа: `first party` и `third party` интеграции.

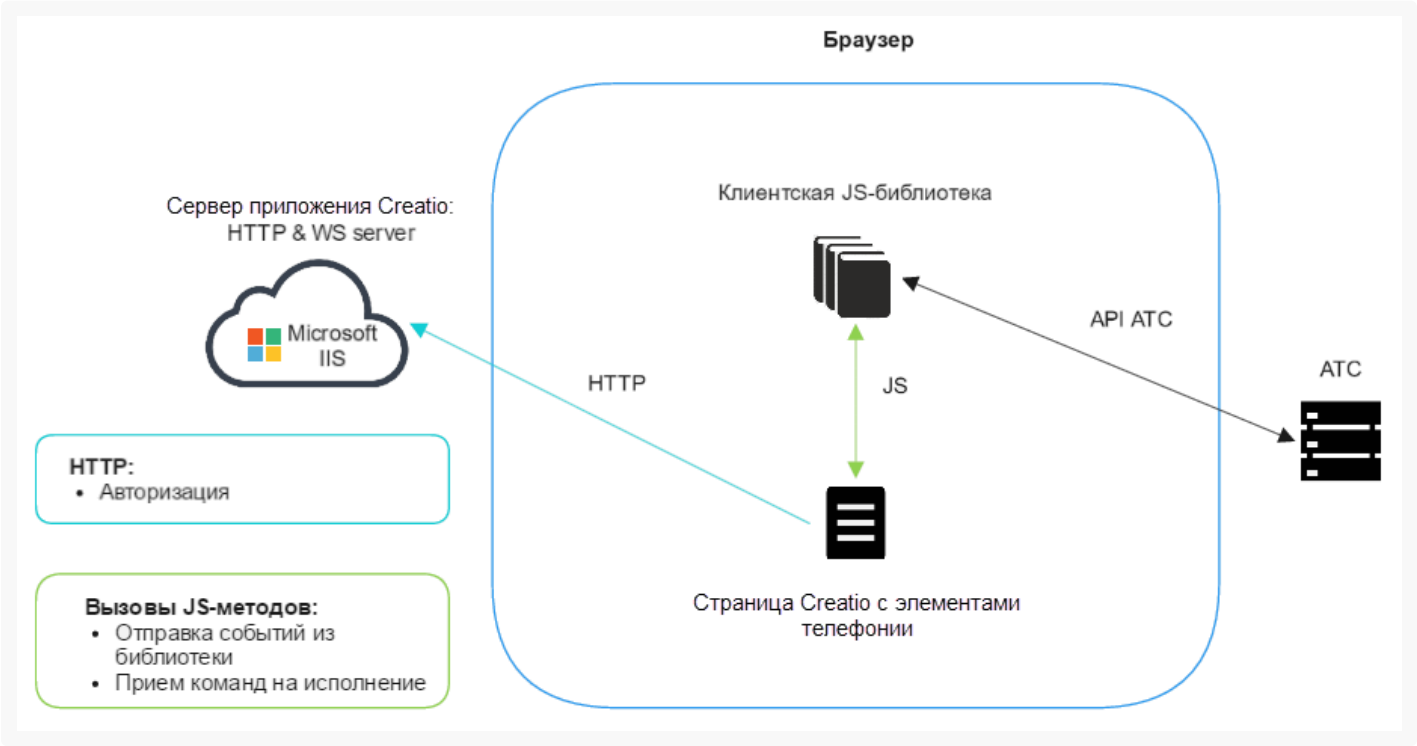
В случае `first party` интеграции для каждого пользователя создается отдельное интеграционное подключение, в рамках которого выполняется обработка событий АТС.

Для `third party` интеграций выполняется одно подключение к серверу АТС и в рамках него выполняется обработка событий АТС для всех пользователей интеграции. В случае third party интеграций применяется промежуточное звено `Messaging Service` для распределения информационных потоков разных пользователей.

JavaScript-адаптер на стороне клиента

При способе интеграции с JavaScript-адаптером на стороне клиента, работа с АТС происходит

непосредственно из браузера. Взаимодействие с АТС и JavaScript-библиотекой, которая, как правило, поставляется производителем АТС, происходит с помощью API АТС. Библиотека отправляет события и принимает команды на исполнение, используя JavaScript. В контексте данной интеграции страница Creatio взаимодействует с сервером приложения для авторизации, используя протокол HTTP(S).



Данный способ интеграции применим к `first party` API телефонии, например, для коннекторов Webitel, Oktell, Finesse. Для коннекторов Webitel и Oktell в качестве протокола соединения используется WebSocket, а для коннектора Finesse используются long-polling http-запросы.

Преимуществом метода интеграции `first party` является тот факт, что для него не требуется наличие дополнительных узлов, например, Messaging Service. CTI-панель с использованием интеграционной библиотеки выполняет подключения напрямую к API сервера телефонии из браузера на ПК пользователя.

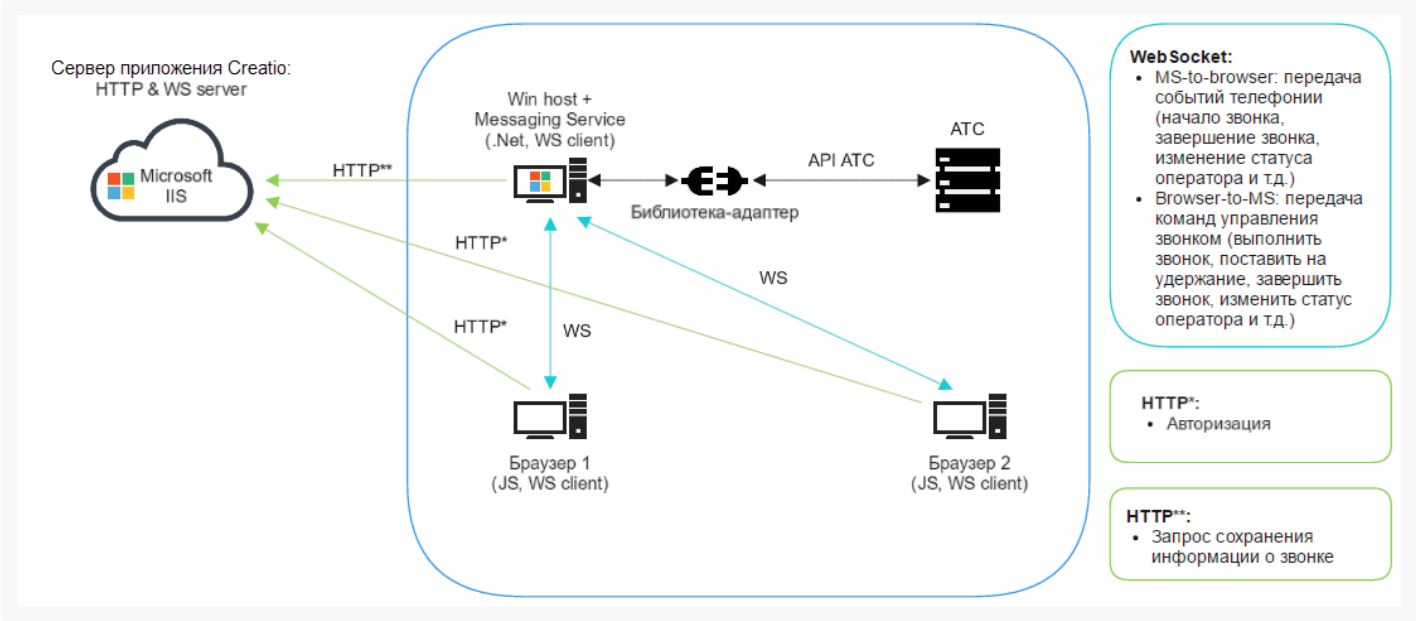
В случае с входящим звонком, сервер телефонии передает событие начала нового звонка и его параметры через WebSocket в библиотеку клиентской интеграции. Библиотека при получении команды нового звонка генерирует событие `RingStarted`, которое передается на страницу приложения.

В случае с исходящим звонком, клиентская часть генерирует команду начала звонка, которая по WebSocket передается на сервер телефонии.

Terrasoft Messaging Service на серверной стороне

При способе интеграции с Terrasoft Messaging Service (TMS) на серверной стороне все события телефонии проходят через данный сервис, который взаимодействует с АТС посредством библиотеки производителя АТС. Библиотека взаимодействует с АТС посредством API. Также TMS взаимодействует с сервером приложения Creatio для запроса сохранения информации о звонке в базе данных, используя HTTP(S). С клиентским приложением взаимодействие (передача событий и прием команд на исполнение) происходит посредством WebSocket. Как и в случае с интеграцией с JavaScript-адаптером на стороне

клиента, страница Creatio взаимодействует с сервером приложения для авторизации, используя протокол HTTP(S).



Данный способ интеграции применим к `third party` API телефонии (TAPI, TSAPI, New Infinity protocol, WebSocket Oktell). Для данного типа интеграции необходим `Messaging Service` — windows proxy-служба, которая работает с библиотекой-адаптером АТС. Messaging Service является универсальным хостером библиотек интеграции с АТС, таких, как Asterisk, Avaya, Callway, Ctios, Infinity, Infra, Tapi. `Messaging Service` при получении клиентских подключений автоматически подключит используемую Creatio библиотеку и инициирует подключение к АТС. Фактически, Messaging Service является функциональной "оберткой" для тех коннекторов телефонии, которые не поддерживают клиентскую интеграцию, для того чтобы взаимодействовать с функциональностью телефонии в браузере (генерация и обработка событий, передача данных). Компьютер пользователя производит два вида коммуникации:

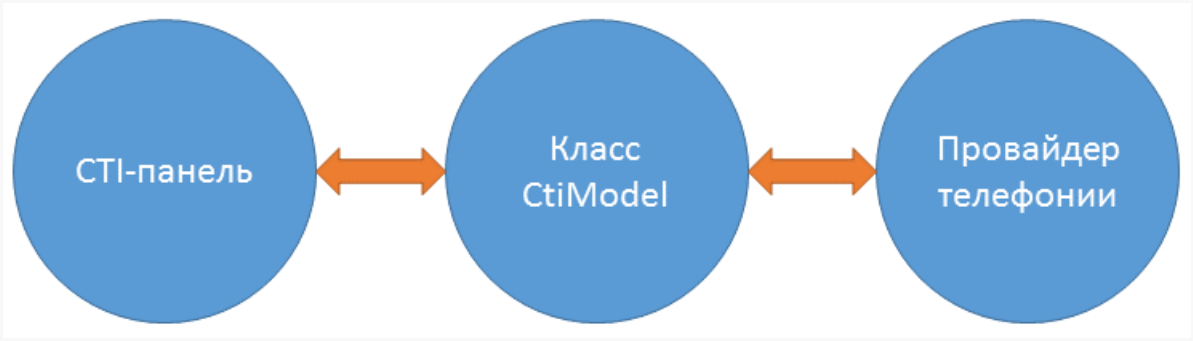
- по протоколу HTTP с сервером приложения Creatio для авторизации и с хостом, на котором установлен Messaging Service;
- по WebSocket для непосредственной работы с функциональностью телефонии.

В случае с входящим звонком, АТС передает событие начала нового звонка и его параметры через библиотеку-адаптер на хост с Messaging Service. Messaging Service при получении команды нового звонка генерирует событие `RingStarted`, которое передается на клиент.

В случае с исходящим звонком, клиентская часть генерирует команду начала звонка, которая по WebSocket передается в Messaging Service, который, в свою очередь, генерирует событие исходящего звонка для АТС.

Взаимодействие коннекторов с Creatio

Все коннекторы взаимодействуют с конфигурацией через класс CtiModel. Он получает события от коннектора и обрабатывает их.



Список поддерживаемых событий класса перечислен в таблице.

Список поддерживаемых событий класса CtiModel

Событие	Описание
initialized	Срабатывает при завершении инициализации провайдера.
disconnected	Срабатывает при отключении провайдера.
callStarted	Срабатывает при начале нового вызова.
callFinished	Срабатывает после завершения звонка.
commutationStarted	Срабатывает после установки соединения звонка.
callBusy	Срабатывает при переводе звонка в состояние "занят" (только TAPI).
hold	Срабатывает после постановки звонка на удержание.
unhold	Срабатывает после снятия звонка с удержания.
error	Срабатывает при возникновении ошибки.
lineStateChanged	Срабатывает после изменения набора доступных операций линии либо звонка.
agentStateChanged	Срабатывает после изменения состояния агента.
activeCallSnapshot	Срабатывает при актуализации списка активных звонков.
callSaved	Срабатывает после создания или обновления звонка в базе данных.
rawMessage	Обобщенное событие в провайдере. Срабатывает при любом событии провайдера.
currentCallChanged	Срабатывает при изменении основного звонка. Например, завершается основной звонок при консультации.

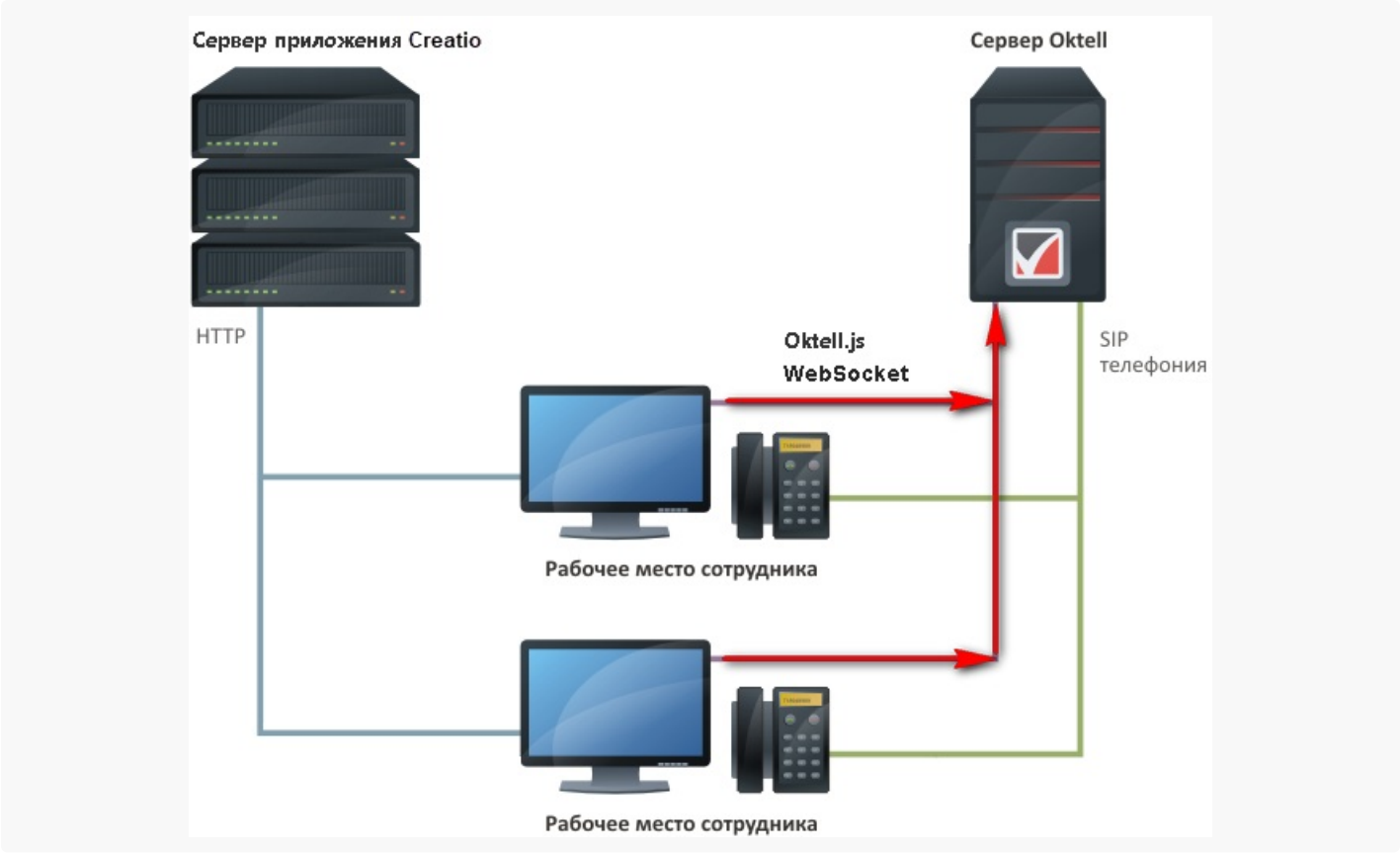
callCentreStateChanged	Срабатывает при входе или выходе оператора из режима Call-центр.
callInfoChanged	Срабатывает при изменении данных звонка по идентификатору в БД.
dtmfEntered	Срабатывает, если в линию были отправлены сигналы Dtmf.
webRtcStarted	Срабатывает при старте webRtc сессии.
webRtcVideoStarted	Срабатывает при старте видеопотока webRtc сессии.
webRtcDestroyed	Срабатывает при завершении webRtc сессии.

Интеграция с Oktell

Сложный

Интеграция Oktell с Creatio реализована на клиентском уровне с помощью библиотеки `oktell.js`. Исходный код библиотеки `oktell.js` находится в конфигурационной схеме `OktellModule` пакета `CTIBase`.

Сервер Oktell взаимодействует с телефонами и с конечными клиентами (браузерами). При таком способе интеграции в Creatio не требуется наличие собственного WebSocket-сервера. Каждый клиент подключается по WebSocket-протоколу непосредственно к серверу Oktell. Сервер приложения Creatio занимается формированием страниц и предоставлением данных из базы данных приложения. Непосредственная взаимосвязь между серверами Creatio и Oktell отсутствует, доступ не требуется, клиенты самостоятельно обрабатывают и объединяют данные двух систем. По такому принципу реализованы Web-клиент Oktell и плагин `oktell.js`, доступный для встраивания в другие проекты.



Oktell.js

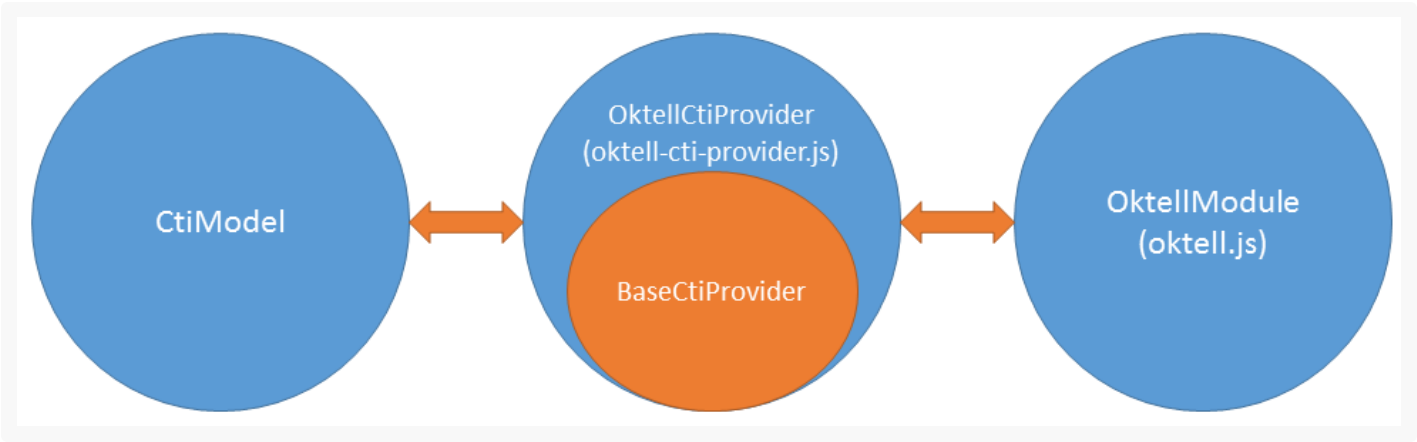
`Oktell.js` — javascript-библиотека для встраивания функциональности управления звонками в CRM-систему. `Oktell.js` использует протокол Oktell WebSocket для соединения с сервером Oktell. Преимущество этого протокола заключается в создании постоянного асинхронного соединения с сервером, которое позволяет без задержек получать события с сервера Oktell и выполнять определенные команды. Поскольку протокол Oktell WebSocket достаточно сложен для реализации, библиотека `Oktell.js` оборачивает внутри себя методы WebSocket-протокола, предоставляя простую функциональность для управления.

Передача голоса между абонентами

Голос при разговоре между операторами oktell Creatio передается по протоколу Session Initiation Protocol (SIP). Для этого требуется использование либо IP-телефона, либо установленного на компьютер оператора софтфона.

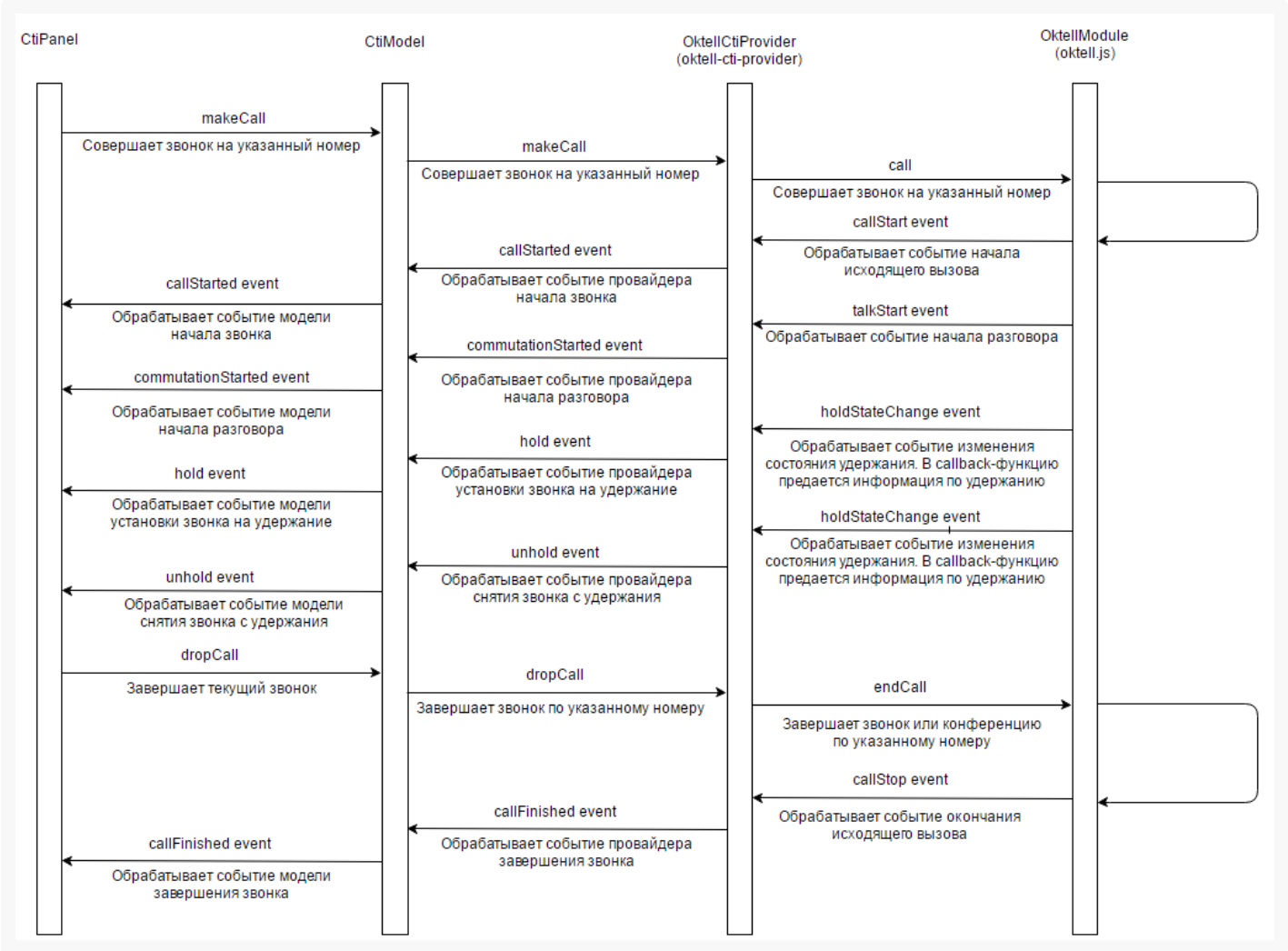
Взаимодействие компонентов

Взаимодействие с библиотекой `oktell.js` осуществляется с помощью класса `OktellCtiProvider`, который является связующим звеном между `CtiModel` и `OktellModule`, в котором находится код `oktell.js`. Класс `OktellCtiProvider` реализует интерфейс класса `BaseCtiProvider`.

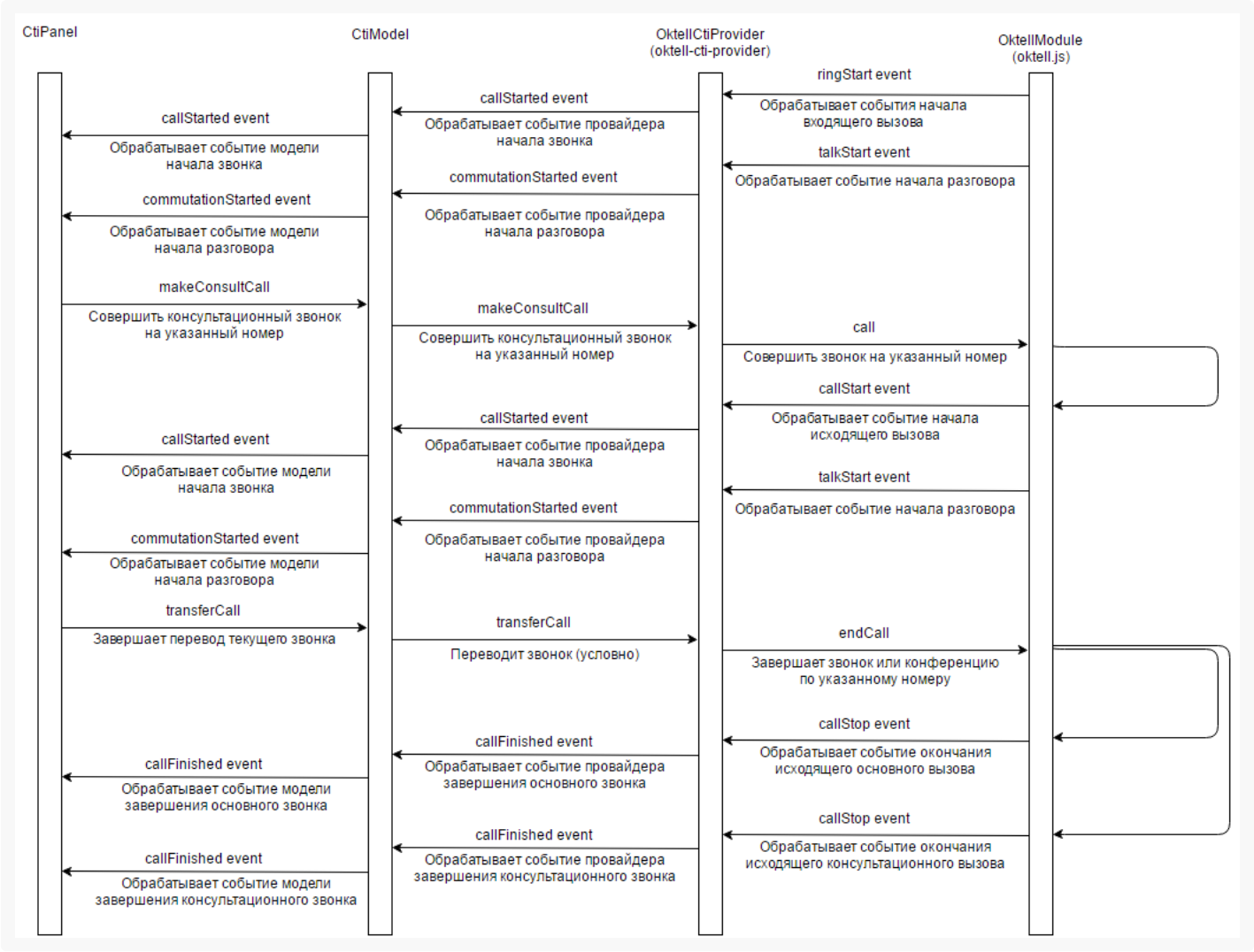


Примеры взаимодействия CtiModel, OktellCtiProvider и OktlellModule:

Исходящий звонок оператора абоненту с установкой звонка на удержание абонентом, снятием с удержания абонентом и завершением звонка оператором



Входящий звонок абонента 1 оператору с консультационным звонком абоненту 2, с последующим соединением оператором абонента 1 и абонента 2



Список поддерживаемых событий класса библиотеки `oktell.js` перечислен в таблице.

Список поддерживаемых событий класса библиотеки `oktell.js`

Событие	Описание
connect	Событие успешного соединения с сервером
connectError	Событие ошибки соединения с сервером в методе connect. Коды ошибок такие же, как для callback-функции метода connect
disconnect	Событие закрытия соединения с сервером. В callback-функцию передается объект с описанием причины разрыва соединения
statusChange	Событие изменения состояния агента. В callback-функцию передается два строковых параметра — новое и прошлое состояние
ringStart	Событие начала входящего вызова
ringStop	Событие завершения входящего вызова
backRingStart	Событие начала обратного вызова
backRingStop	Событие завершения обратного вызова
callStart	Событие начала исходящего вызова
callStop	Событие смены UUID звонка
talkStart	Событие начала разговора
talkStop	Событие окончания разговора
holdAbonentLeave	Событие выхода абонента из удержания. В callback-функцию передается объект abonent с информацией по абоненту
holdAbonentEnter	Событие входа абонента в удержание. В callback-функцию передается объект abonent с информацией по абоненту
holdStateChange	Событие изменения состояния удержания. В callback-функцию передается информация по удержанию
stateChange	Событие изменения состояния линии
abonentsChange	Событие изменения списка текущих абонентов
flashstatechanged	Низкоуровневое событие изменения состояния удержания
userstatechanged	Низкоуровневое событие изменения состояния пользователя

Интеграция с Webitel

Сложный

Интеграция Webitel реализована на клиенте в виде отдельных модулей Creatio. Состав модулей, входящих в интеграцию:

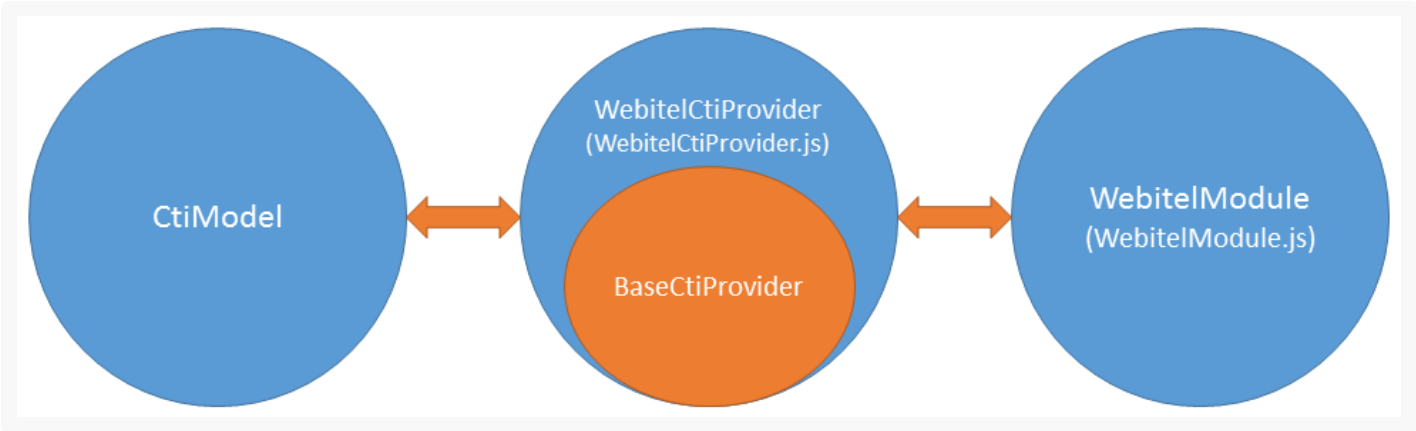
Пакет `WebitelCore` — модули, содержащие низкоуровневые взаимодействия с АТС Webitel с использованием модуля Verto, а также CTI-панели на странице приложения Creatio.

Пакет `WebitelCollaborations` — реализация базовых интерфейсов для работы с Webitel в Creatio. Содержит модуль `WebitelCtiProvider`, реализующий класс `WebitelCtiProvider,` коннектор к Webitel, страницу настройки параметров подключения, справочник для редактирования пользователей Webitel непосредственно в Creatio, необходимые привязки данных.

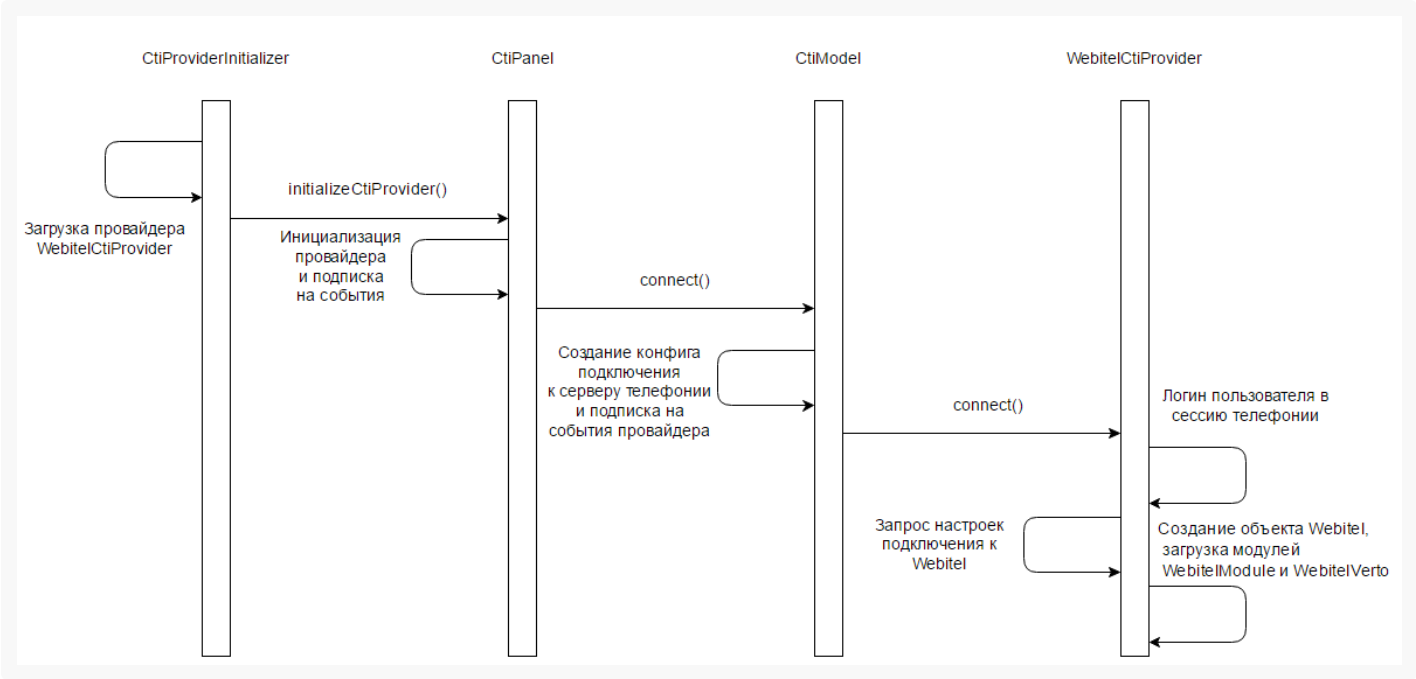
Подробно с архитектурой платформы Webitel можно ознакомиться в документации.

Взаимодействие компонентов

Класс `WebitelCtiProvider` (наследник класса `Terrasoft.BaseCtiProvider`) реализует необходимое взаимодействие между `CtiModel` и низкоуровневым глобальным объектом `Webitel` (модуль `WebitelCore.WebitelModule.js`).



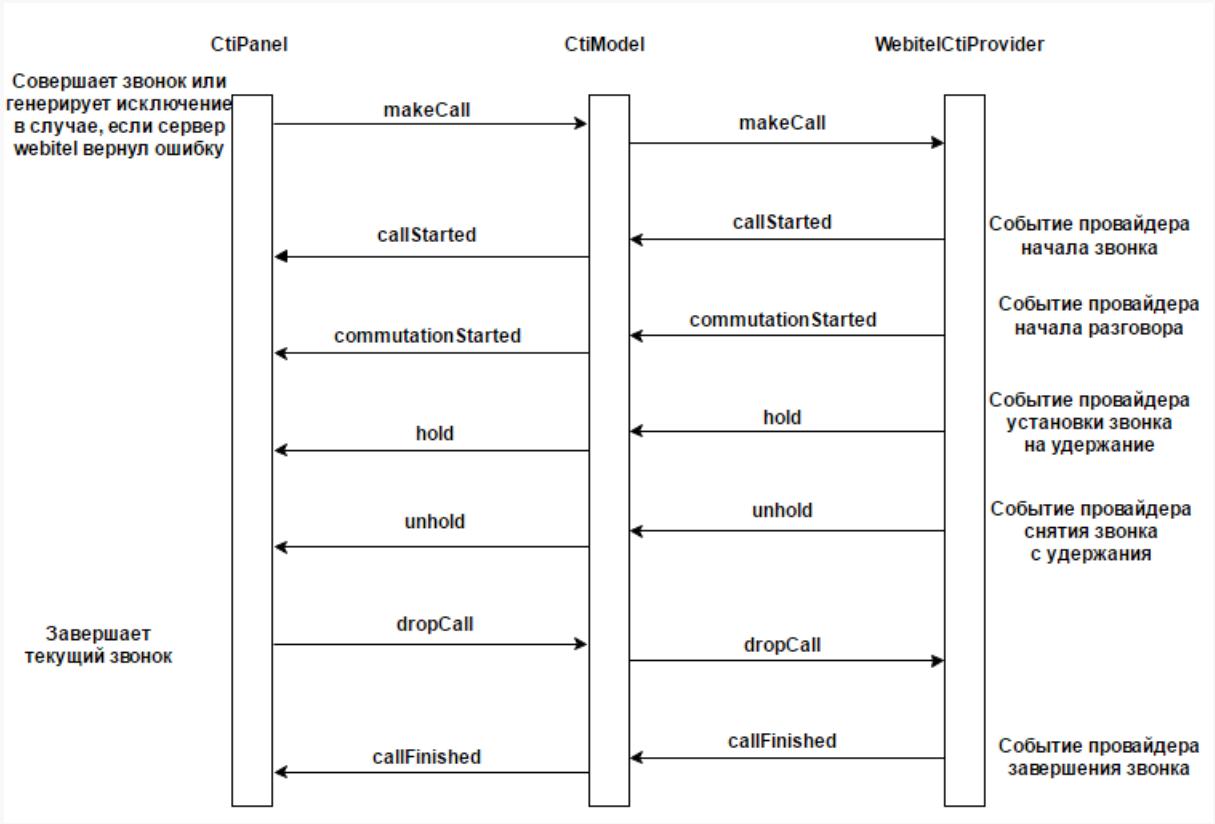
Интеграция осуществляется следующим образом. Если пользователь установил системную настройку библиотеки интеграции Webitel, `CtiProviderInitializer` загружает модуль `WebitelCtiProvider`. Далее вызывается метод `init` в `WebitelCtiProvider`, который осуществляет логин пользователя в сессию телефонии (метод `LogInMsgServer` сервиса `MsgUtilService.svc`). Если логин был успешным, вызывается метод `connect`, который проверяет, что еще нет существующего подключения (свойство `this.isConnected` имеет значение `false` и `this.webitel` — пустое). После этого `connect` запрашивает настройки подключения к Webitel, которые хранятся в системных настройках `webitelConnectionString` и `webitelWebrtcConnectionString`.



После получения системных настроек происходит получение настроек пользователя из справочника [*Пользователи Webitel*], используя метод `GetUserConnection` клиентского сервиса `WUserConnectionService`. После получения пользовательских настроек загружается модуль `WebitelModule` и `WebitelVerto,` если в настройках пользователя установлен признак [*Use web phone*]. Затем вызывается метод `onConnected`, в котором создается глобальный объект `Webitel`, где свойства заполняются настройками подключения. Происходит подписка на события объекта `Webitel` и вызывается метод `connect`, который осуществляет подключение по WebSocket, авторизацию в АТС Webitel и остальные низкоуровневые операции по подключению. При возникновении события `onConnect` подключение считается успешным, и у пользователя появляется возможность работать со звонками. Во время работы коннектора `WebitelCtiProvider` реагирует на события объекта `Webitel`, обрабатывает их и, при необходимости, генерирует события коннектора, описанные в классе `Terrasoft.BaseCtiProvider`. Для управления звонками `WebitelCtiProvider` реализует абстрактные методы класса `Terrasoft.BaseCtiProvider`, используя методы объекта `Webitel`.

Примеры взаимодействия CtiPanel, CtiModel и WebitelCtiProvider

Исходящий звонок оператора абоненту с установкой звонка на удержание абонентом, снятия с удержания абонентом и завершения звонка оператором.



Список портов Webitel

- 871 — порт WebSocket для подключения к серверу Webitel и получения событий.
- 5060 и 5080 — сигнальные порты для подключениz SIP-телефонов и провайдеров телефонии.
- 5066 — порт для подключения Web-телефона, сигнальный порт WebRTC.
- 4004 — порт для получения записей разговоров.

События Webitel

События WebitelCtiProvider

Событие	Описание
onNewCall	Событие начала нового звонка.
onAcceptCall	Событие поднятия трубки.
onHoldCall	Событие удержания звонка.
onUnholdCall	Событие снятия звонка с удержания.
onDtmfCall	Событие тонового набора.
onBridgeCall	Событие соединения с каналом.
onUuidCall	Событие смены UUID звонка.
onHangupCall	Событие завершения звонка.
onNewWebRTCCall	Событие новой WebRTC-сессии.

Интеграция с Asterisk

Сложный

Для взаимодействия с сервером Asterisk используется AMI (Asterisk Manager API). Manager API позволяет клиентским программам соединяться с сервером Asterisk, используя TCP/IP протокол. Данный API (Application Programming Interface) позволяет считывать события, происходящие в автоматической телефонной станции (АТС), и отправлять команды управления звонком.

На заметку. Приложение поддерживает интеграцию с Asterisk версии 13.

Для коммуникации между АТС Asterisk и подсоединенным Manager API клиентом используется простой текстовый построчный протокол вида: "параметр: значение". Окончание строки определяется последовательностью перевода строки и возврата каретки (CRLF).

На заметку. В дальнейшем для набора строк вида "параметр: значение", после которых идет пустая строка, содержащая только CRLF, для упрощения будет использоваться термин "пакет".

Настройка конфигурационного файла сервиса Messaging Service для интеграции Creatio с Asterisk

Для работы интеграции с Creatio необходимо установить Terrasoft Messaging Service (TMS) на выделенном

компьютере, который будет использоваться в качестве сервера данной интеграции. В конфигурационном файле `Terrasoft.Messaging.Service.exe.config` необходимо установить следующие параметры для коннектора Asterisk.

Установка параметров для коннектора Asterisk

```
<asterisk filePath="" url="Имя_или_адрес_сервера_Asterisk" port="Порт_сервера_Asterisk" userName
```

Порты для интеграции Creatio с Asterisk

- TMS принимает от браузера WebSocket подключение на порт 2013 по протоколу TCP.
- TMS подключается к серверу Asterisk по умолчанию на порт 5038.

Сервис Terrasoft Messaging Service для интеграции Creatio с Asterisk

Интеграционная часть Messaging Service реализована в ядре основного решения Creatio в библиотеке `Terrasoft.Messaging.Asterisk`.

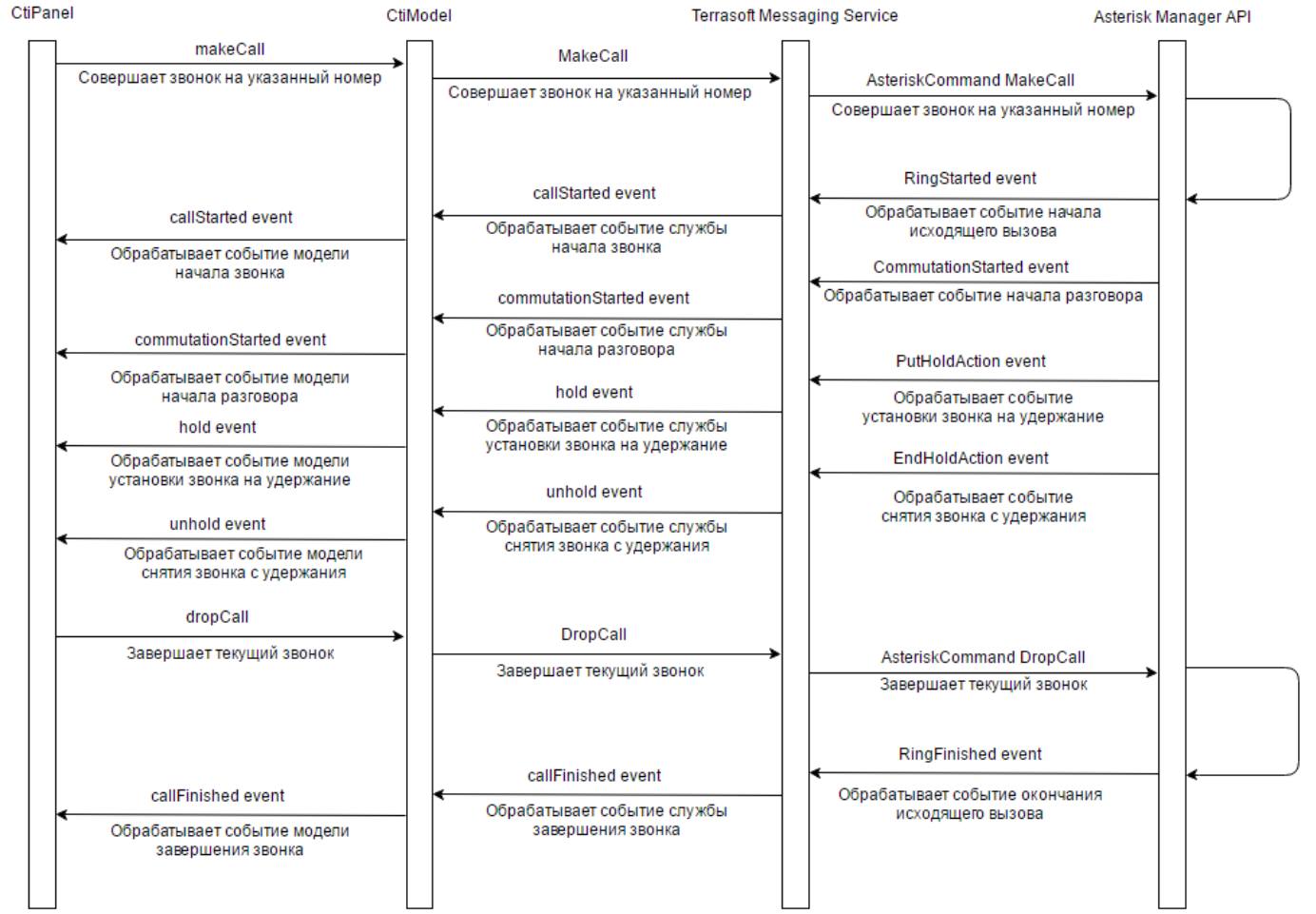
Основные классы библиотеки:

- `AsteriskAdapter` — класс, преобразующий события Asterisk в высокоуровневые события модели звонка, используемой в интеграции Creatio.
- `AsteriskManager` — класс, использующийся для создания и удаления пользовательского соединения с сервером Asterisk.
- `AsteriskConnection` — класс, представляющий собой пользовательское соединение при интеграции с Asterisk.
- `AsteriskClient` — класс, использующийся для отправки команд на сервер Asterisk.

Пример взаимодействия CtiModel, Terrasoft Messaging Service и Asterisk Manager API

Исходящий звонок оператора абоненту с установкой звонка на удержание абонентом, снятия с удержания абонентом и завершения звонка оператором.

Последовательнось возникновения событий при звонке для данного примера:



В таблице приведен пример обработки событий — как данные события интерпретируются TMS, какие значения из приведенных событий используются при обработке входящего звонка.

События Asterisk

Asterisk log	TMS	Action	C
`{ Event: newchannel Channel: <channel_name> UniqueID: <unique_id> }`	Создается канал и добавляется в коллекцию `new AsteriskChannel({ Name: <channel_name> UniqueId: <unique_id> });`		
`{ Event: Hold UniqueID: <unique_id>`	В коллекции каналов ищется канал по `<unique_id>` и с помощью метода `fireEvent` генерируется событие	`PutHoldAction`	C c C)

`Status: "On"` `}`			

`{`
`Event: Hold`
`UniqueID: <unique_id>`
`Status: "Off"`
`}`
	В коллекции каналов ищется канал по `<unique_id>` и с помощью метода `fireEvent` генерируется событие	`EndHoldAction`	

`{`
`Event: Hangup`
`UniqueID: <unique_id>`
`}`
	В коллекции каналов ищется канал по `<unique_id>` и с помощью метода `fireEvent` генерируется событие	`RingFinished`	

`{`
`Event: Dial`
`SubEvent: Begin`
`UniqueID: <unique_id>`
`}`
	В коллекции каналов ищется канал по `<unique_id>`, заполняются данные и с помощью метода `fireEvent` генерируется событие	`RingStarted`	

`{`
`Event: Bridge`
`UniqueID: <unique_id>`
`}`
	В коллекции каналов ищется канал по `<unique_id>` и с помощью метода `fireEvent` генерируется событие	`CommutationStarted`	

События Asterisk

Подробный список событий и информация об их параметрах приведены в документации Asterisk.