|  |  |
| --- | --- |
| **Тип документа** | Документация |
| **Проект** | «Контакт Центр» |
| **Разработчик** | Алексей Алешин |
| **Среда разработки** | PHP7.4 |
| **Стек технологий** | Asterisk18.8.0, MS SQL 2019 EE |
| **Используемые библиотеки** | phpAGI  http://www.voip-info.org/wiki-Asterisk+agi |
| **Цель разработки** | Разработка классов обработки входящих, исходящих, внутренних звонков и переадресаций из АТС Астериск и передачи данных в MS SQL |
| **Сроки выполнения** | С 20.01.22 до 16.02.22 |
| **Количество классов** | 8 |
| **Программный код (исходники)** | http://gitlab.local.com/a.aleshin/event-astreisk |
| **Сервер разработки** | 192.168.0.201 |

**Оглавление**

[Принцип работы. 2](#__RefHeading___Toc6868_60634147)

[Список и описание классов. 2](#__RefHeading___Toc6870_60634147)

[Описание используемых данных в событиях AMI. Все данные формата string. На входе данные полученные от Астериск в виде массива ($data). 3](#__RefHeading___Toc6872_60634147)

[Установка и настройка 4](#__RefHeading___Toc6874_60634147)

### Принцип работы.

При получение события \*\*Newchannel\*\* из Астериск AMI, получаем из него Linkedid и Uniqueid и время поступления вызова. заносим в таблицу из события \*\*Newexten\*\* получаем путь и имя аудио файла, так же заносим в таблицу.

По событию DialEnd - получаем кто и во сколько принял звонок.

При безусловном переводе Blindtransfer - получаем кто и на кого перевел звонок. При переводе с ответом оператора AttendedTransfer - получаем те же параметры, но при этом возникает новый канал, его удаляем из таблицы.

По событию Hangup - получаем значения когда положили трубку.

Каждый класс находиться в отдельном файле с названием события, класс имеет название события и стартовую функцию start() с передачей в нее данных полученных от астериск в массиве $data. Каждый класс заносит данные самостоятельно.

### Список и описание классов.

|  |  |  |  |
| --- | --- | --- | --- |
| п/п | Имя класса | Входные данные | Выходные данные |
| 1 | Eventcall | Слушает порт Астериск, получает от него События | Перенаправляет на нужный класс в зависимости от полученного События, передает событие в массиве $data |
| 2 | Newchannel | Получает на входе Событие Newchannel, образование нового канала при входящем и исходящем звонке | Получаем данные Exten, Linkedid, in\_date, Uniqueid, SRC\_NUM (Дату и время звонка, номер звонящего, номера каналов) |
| 3 | Newexten | Событие Newexten, смена статуса канала. | Получаем данные по файлу аудиозаписи (MixMonitor),путь и название. |
| 4 | Dialend | Событие Dialend, поднятие трубки, занято, отказано | Получаем данные о времени ответа, кто ответил. (up\_date, DET\_NUM). |
| 5 | Blindtransfer | Событие Blindtransfer, безусловная переадресация звонка. | Получаем данные кому переадресовали звонок (transfer\_to, transfer\_from) |
| 6 | Attendedtransfer | Событие Attendedtransfer, перевода звонка с консультацией кому переведен звонок. | Получаем данные кому переведен звонок (transfer\_to, transfer\_from) |
| 7 | Hangup | Событие Hangup, окончание звонка. | Получаем данные о времени завершения звонка, кто положил трубку. |
| 8 | PhoneNumTransfer | Получает на входе номер телефона. | Выдает номер телефона с удаленными лишними знаками (+, -, ,) |

### Описание используемых данных в событиях AMI. Все данные формата string. На входе данные полученные от Астериск в виде массива ($data).

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | **Newchannel** | Channel | Номер входящего номера (внутренний) если не определяется то смотрим внешний номер **CallerIDNum** |
|  |  | CallerIDNum | Номер внешнего входящего звонка. |
|  |  | ChannelState | Статус канала (**4**) активен, ждет поднятие трубки |
|  |  | Uniqueid | Идентификатор канала |
|  |  | Linkedid | Уникальный идентификатор канала, неизменный с начала существования канала |
| 2 | **Newexten** | Linkedid | По данному идентификатору находим запись в таблицы |
|  |  | Application | Если получаем значение “**MixMonitor**”, записываем путь и название аудио файл |
|  |  | AppData | Путь и название аудио файла |
| 3 | **DialEnd** | **Linkedid** | По данному идентификатору находим запись в таблицы |
|  |  | DestContext | “**from-queue**” входящий внешний звонок  “**from-internal**” внутренний звонок |
|  |  | Context | “**macro-dialout-trunk**” исходящий внешний вызов |
|  |  | DialStatus | “**ANSWER**” ответ на звонок – заносим в таблицу дату, время и номер кто ответил.  “**BUSY**” - занято |
|  |  | **ConnectedLineNum** | Номер кто ответил на звонок |
| 4 | **Blindtransfer** | **TransfererLinkedid** | Идентификатор канала который переводится, сравниваем в таблицы с **Linkedid** |
|  |  | **TransfererChannel** | Номер кто переводит |
|  |  | **Extension** | Номер кому переводят |
| 5 | **AttendedTransfer** | **OrigTrannsfererLinkedid** | Идентификатор канала который переводится, ищем в таблице и вносим данные |
|  |  | **SecondTransfererCallerIDNum** | Номер кто переводит |
|  |  | **SecondTransfererConnectedLineNum** | Номер кому переводят |
| 6 | **Hangup** | **Linkedid** | Идентификатор канала по которому ищем в таблице и заносим данные |
|  |  | **Context** | Если “**macro-dial-one**”, “**from-queue**”, “**from-internal**” вносим **Channel**, во всех остальных **CallerIDNum** |
|  |  | **Channel** | Положил трубку внутренний абонент |
|  |  | **CallerIDNum** | Положил трубку внешний абонент |

### Установка и настройка

1. Перенести файлы проекта на сервер.

2. Внести изменения в файлы конфигурации в настройки соединений

Conn\_db.php

|  |
| --- |
| $server = ;  $userName = ;  $password = ;  $queryTimeout = ; |

и Eventcall.php

|  |
| --- |
| $res=$as->connect('127.0.0.1','login', 'pass'); |

3. Запустить фйл php ./Eventcall.php