ООО "РТ МИС"

ЕДИНАЯ ЦИФРОВАЯ ПЛАТФОРМА.МИС 3.0

(ЕЦП.МИС 3.0)

Инструкция по развертыванию

Подсистемы "Голосовой помощник"

Содержание

1 Голосовой помощник

1.1 Требования к полигону развертывания

№	Лицензия / иные документы, ссылка на условия лицензии	Наименование библиотеки/ программы и ссылка на размещение (при наличии)	Фактический способ использования
1	PostgreSQL License	PostgreSQL	Database
4	Apache License 2.0 license	gRPC	Implement gRPC protocol
5	BSD License	Protocol Buffers	Part of gRPC
6	Apache License 2.0	Google Cloud Speech API client library	For speech recognition and synthesis
7	Apache License 2.0	Requests	For sending HTTP *requests*
8	Apache License 2.0	aiohttp	Async http framework
9	BSD or MIT License	Redis	Database
10	BSD license	Python	General program language
11	BSD license	Flask	Http framework
12	MIT license	SQLAlchemy	SQL toolkit and Object Relational Mapper
13	BSD license	Jinja2	*T*emplating engine
14	BSD 3-Clause license	Werkzeug	Part of Flask
15	BSD 3-Clause license	Celery	Task queue implementation for Python
17	MIT license	pydub	Audio processing library
18	MIT license	xlutils	Package for working with Excel files
19	BSD license	lxml	Package for parse xml
20	BSD license	passlib	Password hashing library
21	MIT license	pickle	For serialize and deserialize objects
22	BSD license	scipy	For wav file read
23	BSD license	sklearn	For learn Gaussian Mixture Models

24	MIT license	python_speech_features	Speech processing
25	BSD license	numpy	Computation framework
26	MIT license	bootstrap	front-end open source toolkit for develop html pages
27	MIT license	jquery	JavaScript library for HTML DOM tree traversal and manipulation
28	MIT license	datatables	Plug-in for the *jQuery* Javascript library, for create tables
30	MIT license	Redis	Database
31	MIT license	urllib	For sending HTTP *requests*
32	Apache License 2.0	tensorflow	For collection of workflows to develop and train models using *Python*
33	MIT license	tflearn	Transparent deep learning library built on top of Tensorflow
34	MIT license	Keras	Deep learning framework
35	Apache License 2.0	PlaidML	Deep learning framework
36	ISC license	Librosa	For music and audio analysis

1.1.1 Требования к техническим и программным средствам

Функциональные характеристики

№ п.	Характеристика
1.	Наличие графического интерфейса на русском и английском языках
2.	Возможность обзванивать абонентов по предоставленным спискам. В списках помимо номера телефона могут быть дополнительные параметры, используемые в скрипте
3.	Возможность установки разрешенного времени суток, интервалов и дней недели для исходящего звонка с учетом часовых поясов
4.	Возможность установки для исходящих вызовов количества попыток дозвона каждому контакту
5.	Возможность определения успешного дозвона до абонента, регистрация статуса звонка и запуск сценария только при дозвоне
6.	Возможность определения автоответчика или голосовой почты
7.	Возможность настройки перезвонов абоненту в настраиваемый промежуток времени с настраиваемым количеством попыток дозвона

8.	Возможность реагировать с минимальной задержкой, соответствующей естественной задержки человеческой речи
9.	Возможность записывать аудиозапись и транскрипцию всех разговоров (в т.ч. незавершенных) в нужном Заказчику стерео-формате и отправлять в согласованное хранилище
10.	Возможность хранения записей
11.	Возможность формирования статистики по всем проведенным звонкам в формате xls или csv и отправлять ее согласованным способом в согласованное время
12.	Возможность определения просьбы абонента перезвонить в определенный промежуток времени и перезванивать абоненту в этот промежуток времени
13.	Возможность сохранять историю контакта с абонентом по сценарию и начать диалог с места обрыва при перезвоне или поступлении входящего звонка от абонента, если это предусматривается сценарием
14.	Возможность перебивания (как в целом, так и на конкретные фразы)
15.	Возможность переводить звонок на оператора, если это предусмотрено сценарием
16.	Возможность определения пола абонента
17.	Наличие встроенного визуального конструктора диалогов
18.	Наличие в сценариях тегов, которые в диалоге заменяются на информацию из базы данных
19.	Возможность для Абонента передавать данные голосом и в DTMF-режиме
20.	Поддержка нелинейной логики с возможностью переключения между ветками диалога без потери контекста
21.	Использование фоновых шумов во время диалога
22.	Использование активного слушания во время диалога
23.	Наличие собственного 2-х уровневого решения NLU (Natural Language Understanding) для обработки текста на естественном языке с целью выделения сущностей и пользовательских намерений, работающего на паттернах на первом уровне и на алгоритмах глубокого обучения – второй уровень NLU
24.	Возможность генерации SMS
25.	Возможность генерации e-mail
26.	Программные интерфейсы для интеграции посредством HTTP API
27.	Поддержка ASR / TTS от Google, Yandex и ЦРТ
28.	Функционал выбора для бота тембра и пола голоса

29.	Возможность распознавать многозначные числа (номера лицевых счетов или показания), которые абонент произносит, произвольно группируя цифры (например, сто тридцать два, пятнадцать, два нуля, шесть)
30.	Возможность распознавать даты, дни недели
31.	Возможность автоматической обработки адресов
32.	Возможность автоматической обработки ФИО
33.	Журналирование и логирование, позволяющее быстро и четко диагностировать возникающие проблемы.
34.	Эксплуатация в режиме 24x7
35.	Возможность установки в изолированном сетевом сегменте
36.	Наличие интерфейса для мониторинга и контроля
37.	Возможность интеграции с Whatsapp
38.	Возможность интеграции с Viber
39.	Возможность интеграции с Telegram
40.	Возможность интеграции с FB Messenger
41.	Возможность интеграции с Skype

Технические особенности

№ п.	Характеристика
1	Микросервисная архитектура
2	СУБД PostgreSQL v.11, нереляционная СУБД MongoDB v.4.2, файловое хранилище S3
3	Платформа располагается на виртуальном облачном хостинге Яндекс.Облако
4	Собственное решение медиа-части, которое обеспечивает масштабируемость, надежность и более широкий функционал по сравнению со стандартным UniMRCP, голосовая платформа собственной разработки (использование FreeSwitch только в качестве SBC), NLU собственной разработки
5	Отказоустойчивость за счет использования виртуального хостинга и резервирования компонент в разных зонах доступности
6	Хранение и передача данных в защищенном виде
7	Механизмы обеспечения защиты данных при обмене: HTTPS, IPsec и др.
8	Возможность горизонтального масштабирования, использование контейнерных технологий
9	Опыт интеграции с Genesys, Mango Office, Kazahtelecom, Avaya и др. Возможна интеграция по API с любой системой клиента через слой middleware. По интеграции с телефонией по SIP-протоколу мы можем интегрироваться с любой системой
10	Соответствие требованиям федерального закона № 152-ФЗ «О персональных данных»
11	Выполнение мер по защите персональных данных согласно Постановлению № 1119 и 21 приказу ФСТЭК в соответствии с требованиями к 3-му уровню защищенности (УЗ-3)

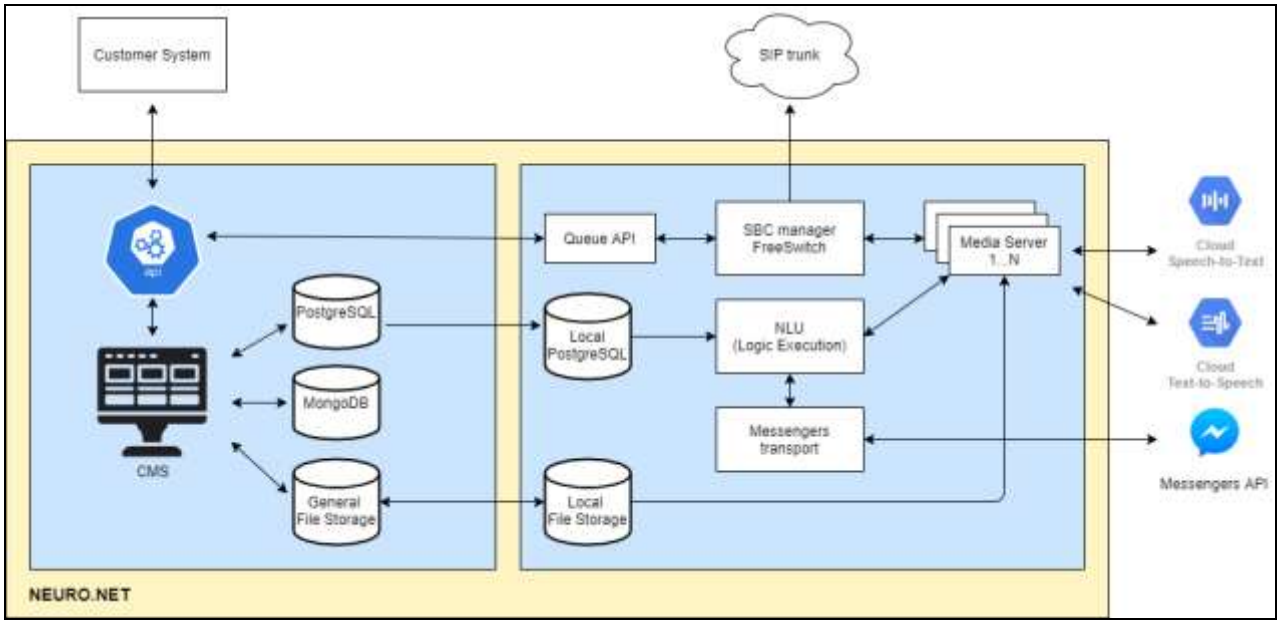
Конфигурация оборудования

В минимальной конфигурации до 20 каналов нужно 4 сервера:

- DB - 4xCPU, 4 Gb RAM, 20 HDD
- Pool API - 4xCPU, 4 Gb RAM, 20 HDD
- Media Server - 4xCPU, 4 Gb RAM
- SBC - 2xCPU, 4Gb RAM

Описание процессов, обеспечивающих поддержание жизненного цикла ПО

High Level Design



Backend Unit

Сервисы для настроек, сбора и отображения статистики по звонкам:

– CMS
– API
– PostgreSQL
– MongoDB
– General File Storage

Может устанавливаться в закрытый контур.

CMS - пользовательский UI:

– управление Conversational flow
– управление агентами
– управление пользователями
– дашбоард
– очереди
– загрузка звонков
– управление аудиофайлами и записями сущностей, диалогами, контактами и т.д.

HTTP API – для бэкенда и интеграции с внешними партнерами.

General File Storage – файлы промптов, записи звонков.

PostgreSQL – база для хранения метаданных: пользователи, агенты, компании. Содержит загруженные звонки, которые мигрируют в пулы, скрипты и прочие пользовательские данные.

MongoDB – хранит статистику, контакты и т.д.

Pool Unit

Это набор сервисов для реализации диалога посредством голоса и текста:

- SBC Manager
- Media Servers
- NLU
- Messengers Transport
- Local File Storage

Может устанавливаться в закрытый контур.

Local PostgreSQL – очередь звонков, статистика звонков до миграции в MongoDB.

Local File Storage – файлы промптов, записи звонков.

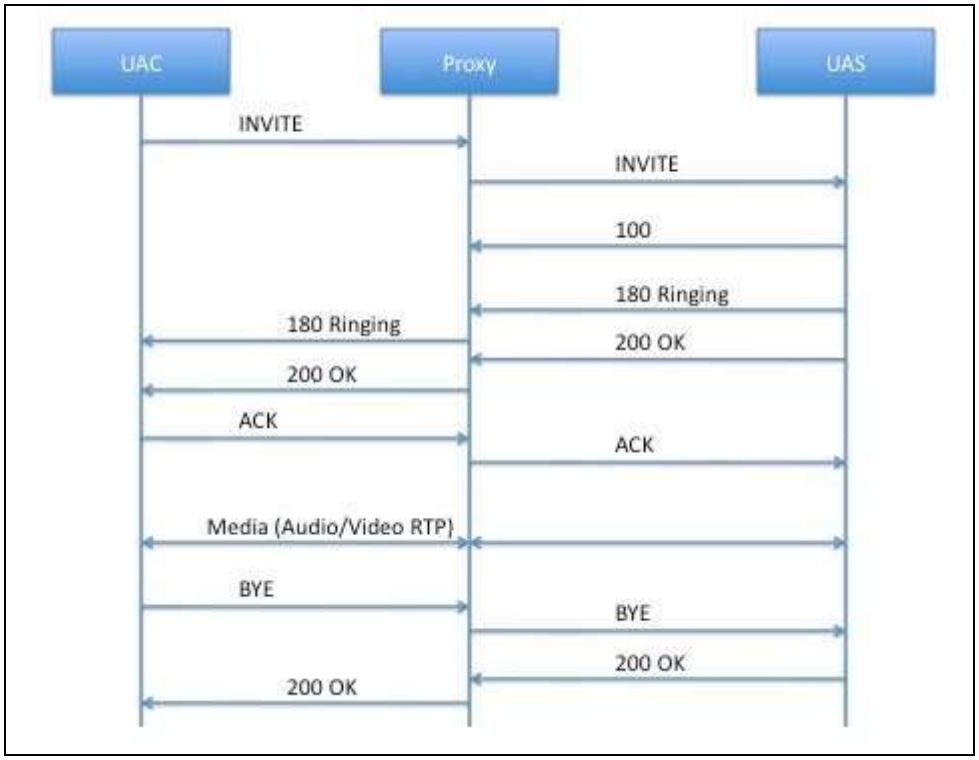
SBC-manager – работает с FreeSWITCH по протоколу ESL, отдает ему команды и получает информацию о каналах.

Media Server – приложение, которое непосредственно работает с SIP и RTP, играет промпты, слушает абонента и интегрируется с ASR.

Logic Executor – выполняет всю логику.

Pool API – работает с очередью звонков, загружает их из БД в память и отдает по запросу.

Call Flow



1.1 Core

Модуль	Описание
Frontend Angular JS	UI системы, написанный на AngularJS.
Backend REST API	REST API для Frontend (ссылка на Swagger), а также external API (ссылка на external API) для клиентов, написанная на Python с использованием фреймворка Flask.
File storage GlusterFS	Файловое хранилище на основе распределённой, параллельной, линейно масштабируемой файловой системы GlusterFS. Служит для хранения аудиозаписей фраз, сущностей и т. д.
MongoDB	Служит для хранения статистики диалогов, биллинга и различных логов.
DB SQL Postgres	Операционная SQL БД, служит для хранения пользователей, паролей, прав пользователей, скриптов, фраз и т.д.

RabbitMQ	Содержит в себе загружаемую через API очередь диалогов. Посредством Dialog distributor распределяет диалоги по пулам, в базу PostgreSQL пула.
Redis	Хранит JWT-токены пользователей.
Middleware	Связующее программное обеспечение для реализации нестандартных сценариев клиента. Имеет свою базу данных (DB SQL Postgres).

1.2 Pool Unit

Модуль	**Описание**
PostgreSQL	*Хранит в себе операционные данные пула, состояние диалогов, временную статистику по диалогам и биллингу, а также несколько логически реплицируемых таблиц (с Postgres Core'a) с информацией по скриптам и аудиозаписям и т.д.*
SBC FreeSWITCH	*Session border controller (SBC), используется для сокрытия топологии сети, перекодирования при обработке вызовов на медиа-сервера.*
SBC manager	*Служит для управления входящими вызовами. Определяет, на какой медиа-сервер можно перевести входящий вызов с* SBC FreeSWITCH.
Media server	*Управляет медиа-потоком, отправляет данные на распознавание, может останавливать звонок, переводить на оператора, делает записи разговоров.*
logic_executor_online	*Процесс, который обрабатывает онлайн-часть диалогов (звонков).*
logic_executor_offline	*Процесс, который обрабатывает оффлайн-*

	часть диалогов.
Caller	*Запускает звонки (проверка лимитов TCL, временных промежутков, проверка доступности положений).*
GlusterFS partition	*Является разделом файловой системы сервера с аудиозаписями фраз, сущностей и т. д.*
S3_migrator	*Скидывает с каждого медиа-сервера записи разговоров в* S3 *хранилища.*
Pool Queue API	*Приложение для управления очередью диалогов и звонков. Занимается их приоритизацией.*
NLU Engine	*Интент-классификатор на основе регулярных выражений. Либо может использовать* NLU Model API *для классификации на основе нейролингвистической модели.*
Message transport	*Собирательное понятие из приложений, которые занимаются отправкой и получением сообщений.*
ASR/TTS	*Сервисы для синтеза и распознавания голоса.*

1.3 Требования по обеспечению безопасности

Система должна выполнять требования законодательства РФ и подзаконных актов в части защиты персональных данных и тайны связи.

Система должна поддерживать интеграцию с имеющимися системами, и не вводить новые системы, дублирующие функционал имеющихся.

Программный продукт должен иметь механизмы проведения идентификации и аутентификации пользователей.

Система должна обеспечивать уровень протоколирования действий сотрудников и клиентов достаточный для разрешения конфликтных ситуаций. Глубина протоколирования должна определяться действующим законодательством и практикой разрешения споров.

Система поддерживает ролевую модель доступа.

Система поддерживает шифрование данных при взаимодействии с другими приложениями или системами.

1.4 Инструкция по установке пользователя на виртуальную машину

На сервере настроен автозапуск компонентов через systemd

ip адрес сервера настраивается автоматически через dhcp

login: root

password: 1q2w3e4r5t

CMS

1. Создать файл */etc/neuro-cms/86a317cb419b.json* с ключами доступа к google speech service

"private_key_id": ""

"private_key": ""

"client_id" : ""

"client_email": ""

/etc/neuro-cms/config.py

Добавляем ключи доступа к облачным сервисам.

SECRET_KEY = «»

#S3 keys

AWS_ACCESS_KEY_ID="

AWS_SECRET_ACCESS_KEY="

YANDEX_OAUTH_TOKEN = "

Распознование речи STC, указываем следующие параметры.

STC_DOMAIN_ID = "

STC_USERNAME = "

STC_PASSWORD = "

MEGAFON_COMPANY_ID = "

2. systemctl start cms.service

API

1. Создать файл с конфигами и прописать ключи доступа */etc/api-neuro/config.py*

SECRET_KEY = ''
#S3 keys
AWS_ACCESS_KEY_ID=''
AWS_SECRET_ACCESS_KEY=''
YANDEX_OAUTH_TOKEN = ''
STC_USERNAME = ''
STC_PASSWORD = ''

2. systemctl start api.service

PostgreSQL

Инстансы PostgreSQL запускаются на портах 5432 и 5433

systemctl start postgresql-10.service
systemctl start postgresql-10-5433.service

MongoDB

Запускается на порту 27017
systemctl start mongod.service

Redis

Запускается как сервис на порту 6379
systemctl status redis.service

MediaServer

1. Caller
systemctl start caller.service

Конфиги настроены и не требуют дополнительных настроек.

/opt/aio-ivr/config.py

/opt/aio-ivr/config.ini

2. Ivr

systemctl start ivr.service

Конфиги настроены и не требуют дополнительных настроек.

/opt/aio-ivr/config.py

opt/aio-ivr/config.ini

3. mrcp-server-google-synth

systemctl start mrcp-server-google-synth.service

В файле /opt/config/neuronet-credentials.json указываем параметры подключения google synth

"type": "service_account",
"project_id": "",
"private_key_id": "",
"private_key": "-----BEGIN PRIVATE KEY-----<you key>-----END PRIVATE KEY-----\n",
"client_email": "",
"client_id": "",
"auth_uri": "https://accounts.google.com/o/oauth2/auth",
"token_uri": "https://oauth2.googleapis.com/token",
"auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
"client_x509_cert_url": ""

4. mrcp-server-yandex-synth

В файле */opt/mrcp-server-yandex-synth/config.ini* указываем параметры подключения к yandex synth

FOLDER_ID = «»

yandex_api_key = «»

OauthToken+OauthUrl = «»

5. mrcp-server

В файле */opt/mrcp-server/config.ini* указываем параметры подключения к yandex

FOLDER_ID = «»

yandex_api_key = «»

OauthToken+OauthUrl = «»

6. mrcp-server-2

systemctl start mrcp-server-2.service

В файле /opt/config/neuronet-credentials.json указываем параметры подключения google synth

"type": "service_account",

"project_id": "",

"private_key_id": "",

"private_key": "-----BEGIN PRIVATE KEY-----<you key>-----END PRIVATE KEY-----\n",

"client_email": "",

"client_id": "",

"auth_uri": "https://accounts.google.com/o/oauth2/auth",

"token_uri": "https://oauth2.googleapis.com/token",

"auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",

"client_x509_cert_url": ""

Freeswitch

1. В /etc/freeswitch/sip_profiles/external.xml добавляем транк до абонента.

2. В /etc/freeswitch/dialplan/dialplan.xml неообходимо настроить bridge до абонента.

<context name="internal">

<extension name="trunk_1">

<condition field="${sip_h_X-Via-Trunk}" expression="^trunk_1$">

<action application="log" data="INFO ->${sip_h_X-Via-Trunk}"/>

<action application="export" data="nolocal:absolute_codec_string=PCMA"/>

<action application="bridge" data="sofia/gateway/trunk_1/${destination_number}"/>

</condition>

</extension>

</context>

queue_api

queue_api сконфигурирован и не требует настроек

call_migrator

call_migrator сконфигурирован и не требует дополнительных настроек