

# Содержание

|   |    |
|---|----|
| Введение . . . . .  | 7  |
| 1 Аналитический раздел . . . . .                              | 8  |
| 1.1 Обоснование задачи . . . . .                              | 8  |
| 1.2 Обзор существующих решений . . . . .                      | 9  |
| 1.2.1 Постановка задачи аутентификации по голосу . . . . .    | 10 |
| 1.2.2 Нормализация входного речевого сигнала . . . . .        | 11 |
| 1.2.3 Выделение характерных признаков . . . . .               | 12 |
| 1.2.4 Построение модели источника речи . . . . .              | 13 |
| 1.2.5 Принятие решения . . . . .                              | 19 |
| 1.3 Выводы . . . . .  | 19 |
| 2 Конструкторский раздел . . . . .                            | 21 |
| 2.1 Общая архитектура разрабатываемого комплекса . . . . .    | 21 |
| 2.2 Варианты использования . . . . .                          | 22 |
| 2.3 Подсистема обучения модели (стадия регистрации) . . . . . | 24 |
| 2.4 Подсистема голосовой аутентификации . . . . .             | 29 |
| 2.5 Подсистема хранения голосовых данных . . . . .            | 32 |
| 2.6 Ядро комплекса голосовой аутентификации . . . . .         | 34 |
| 2.7 Алгоритм удаления тишины . . . . .                        | 42 |
| 2.8 Выводы . . . . .  | 45 |
| 3 Технологический раздел . . . . .                            | 46 |
| 3.1 Выбор среды разработки . . . . .                          | 46 |
| 3.2 Описание программы . . . . .                              | 48 |
| 3.2.1 Общие сведения . . . . .                                | 48 |
| 3.2.2 Функциональное назначение . . . . .                     | 48 |
| 3.2.3 Описание логической структуры . . . . .                 | 49 |
| 3.2.4 Используемые технические средства . . . . .             | 53 |
| 3.2.5 Вызов и загрузка . . . . .                              | 55 |
| 3.3 Руководство пользователя . . . . .                        | 55 |
| 3.3.1 Назначение программы . . . . .                          | 55 |
| 3.3.2 Условия выполнения программы . . . . .                  | 55 |
| 3.3.3 Выполнение программы . . . . .                          | 55 |
| 3.3.3.1 Создание индивидуальной голосовой модели . . . . .    | 56 |

|         |   |    |
|---------|---|----|
| 3.3.3.2 | Аутентификация по голосу . . . . .  | 61 |
| 3.3.3.3 | Интерфейс администратора . . . . .  | 63 |
| 4       | Экспериментальный раздел . . . . .  | 64 |
| 4.1     | Планирование эксперимента . . . . .   | 64 |
| 4.1.1   | Определение точности системы аутентификации . . . . .   | 64 |
| 4.1.2   | Описание экспериментов . . . . .  | 64 |
| 4.1.3   | Описание входных данных . . . . .   | 65 |
| 4.2     | Анализ результатов . . . . .  | 67 |
| 5       | Раздел по охране труда и экологии . . . . .   | 71 |
| 5.1     | Анализ опасных и вредных факторов при разработке программно-<br>го обеспечения и мероприятия по их устранению . . . . . | 71 |
| 5.1.1   | Микроклимат . . . . .   | 71 |
| 5.1.2   | Шум и вибрации . . . . .  | 72 |
| 5.1.3   | Освещение . . . . .   | 73 |
| 5.1.4   | Рентгеновское излучение . . . . .   | 74 |
| 5.1.5   | Неионизирующие электромагнитные излучения . . . . .   | 74 |
| 5.1.6   | Визуальные параметры . . . . .  | 75 |
| 5.2     | Расчет системы искусственного освещения . . . . .   | 76 |
| 6       | Технико-экономический раздел . . . . .  | 81 |
| 6.1     | Организация и планирование процесса разработки . . . . .  | 81 |
| 6.1.1   | Формирование состава выполняемых работ и группировка<br>их по стадиям разработки . . . . .                              | 81 |
| 6.1.2   | Расчет трудоемкости выполнения работ . . . . .  | 82 |
| 6.1.3   | Расчет количества исполнителей . . . . .  | 87 |
| 6.1.4   | Календарный план-график разработки . . . . .  | 89 |
| 6.2     | Определение цены программной продукции . . . . .  | 90 |
| 6.2.1   | Расчет экономической эффективности . . . . .  | 91 |
| 6.3     | Вывод . . . . .   | 92 |
|         | Заключение . . . . .  | 93 |
|         | Список использованных источников . . . . .  | 94 |

## Обозначения и сокращения

- MFCC — Mel-frequency cepstral coefficients. В обработке сигналов, мел-частотные кепстральные коэффициенты представляют краткосрочный спектр мощности звука, основанный на линейном косинусном преобразовании спектра мощности мел-частот;
- GMM — Gaussian mixture model. Модель смеси гауссиан – статистическая модель, позволяющая оценить плотность распределения случайной величины, представляя функцию плотности распределения в виде взвешенной суммы функций нормального распределения (компонент смеси);
- UBM — Universal background model – универсальная фоновая модель. В голосовой аутентификации – гипотетическая модель, отражающая общие характеристики речи для всех возможных источников речи (а также реальные аппроксимации данной модели);
- HMM — Hidden Markov Models – скрытая Марковская модель. Статистическая модель, имитирующая работу процесса, похожего на марковский процесс с неизвестными параметрами, и задачей ставится оценка неизвестных параметров на основе наблюдаемых. Полученные параметры могут быть использованы в дальнейшем анализе, например, для распознавания образов;
- ANN — Artificial Neural Networks – искусственные нейронные сети.

## Введение

Целью данной работы является создание программного комплекса аутентификации пользователя по голосу. В качестве целевой платформы была выбрана система управления студенческими проектами кафедры ИУ7, активно используемая студентами и преподавателями кафедры.

Для достижения данной цели необходимо решить следующие основные задачи:

- разработать схему хранения голосовых данных пользователей с целью их последующего использования для верификации;
- разработать подсистему голосовой верификации, имеющую клиент-серверную архитектуру, позволяющую с заданной вероятностью ошибки определять факт принадлежности речевой последовательности заданному пользователю;
- разработать интуитивный пользовательский интерфейс, позволяющий пользователю передавать на сервер голосовые данные и производить аутентификацию на интернет-ресурсе;
- исследовать разработанный программный комплекс, оценить его работоспособность.

# 1 Аналитический раздел

В данном разделе проводится обоснование поставленной задачи, а также обзор современных подходов к решению проблемы голосовой верификации.

## 1.1 Обоснование задачи

Цель разработчика интерфейсов сделать взаимодействие с пользователем более удобным. Необходимость ввода пары «имя – пароль» при стандартной процедуре аутентификации вынуждает пользователя запоминать сложные численно-буквенные комбинации, или использовать для этого компьютерные программы. Биометрические способы аутентификации, такие как аутентификация по голосу или по отпечаткам пальцев, позволяют не использовать в качестве идентификаторов личности символьные пароли и электронные ключи. Биометрическая аутентификация производится по действительно индивидуальным признакам личности, а не по ассоциированному с личностью материальному носителю или коду. Биометрия практически исключает возможность несанкционированных действий как следствие потери, кражи или передачи пароля третьим лицам. Биометрические способы аутентификации также могут быть использованы вместе с процедурой ввода символьного пароля, увеличивая таким образом степень защиты.

На данный момент, по данным Международного Союза Электросвязи (ITU), более 26% населения планеты имеет доступ в сеть Интернет, по данным исследовательской компании Netcraft (<http://news.netcraft.com/>, 2010) функционирует более 231 миллиона интернет-сайтов, веб-приложения имеют самую большую потенциальную аудиторию. Вместе с этим, в современных веб-приложениях не используются механизмы голосовой аутентификации, поэтому в качестве потенциальной целевой платформы выбрано именно пространство интернет-сайта. При этом появляются следующие ограничения:

- а) разрабатываемый комплекс должен иметь клиент-серверную архитектуру;
- б) должна обеспечиваться интеграция комплекса с остальной инфраструктурой защищаемого интернет-сайта.

Коммерческая емкость рынка биометрических систем постоянно возрастает. Согласно отчетам «Computer Crime», опубликованным Computer Security Institute и Computer Intrusion Squad FBI ([http:](http://)

[//www.computersecurityinstitute.com/](http://www.computersecurityinstitute.com/)) в 2007, 2008 и 2009 годах, за данные три года количество крупных коммерческих фирм и государственных организаций, использующих биометрические системы управления доступом, выросло более чем на 5%. Отчет «Biometric Systems: Worldwide Deployments, Market Drivers, and Major Players» компании Allied Business Intelligence приводит следующие цифры [2]. Объем прибыли от продажи, установки и обслуживания биометрических систем во всем мире вырастет с 302 млн. долларов в 2008 до 476 млн. долларов в 2009 году, т.е. более чем на 34%. В отчете также утверждается, что подобная динамика будет сохраняться до 2011 года. Расширение рынка биометрических систем управления доступом свидетельствует об интересе потребителей к решениям такого рода. Основными пользователями биометрических систем по-прежнему являются крупные государственные учреждения и частные корпорации. В отчете Allied Business Intelligence приводится следующая статистика областей: более 69% потребителей — правительственные, частные корпорации, медицинские организации и т.д. Основными пользователями биометрических систем являются крупные государственные учреждения и частные корпорации.

## **1.2 Обзор существующих решений**

Системы голосовой аутентификации получают все большее распространение в таких сферах, как банковские услуги, при подтверждении сделок, в системах «умный дом» [2]. Основными направлениями применения биометрических систем является биометрическое управление доступом к помещениям (т.н. биометрические замки), а также управление доступом информационным ресурсам компьютерных сетей и рабочих (персональные биометрические системы аутентификации). Однако, аналогов, предоставляющих альтернативу традиционному вводу имени и пароля для входа в защищаемые разделы интернет-порталов, автором найдено не было. Далее в данном разделе дан обзор основных методов, используемых при построении современных систем голосовой аутентификации, для каждой из стадий. Раздел заканчивается краткими выводами и предположениями о направлениях исследований в данной сфере. В обзоре даны описания общих принципов и методов, не учитывая специфики данного проекта, заключающейся в особенностях веб-приложений, а также в целом приложений с клиент-серверной архитектурой.

### 1.2.1 Постановка задачи аутентификации по голосу

Задачи, связанные с определением пользователя по голосу, можно разделить на идентификацию и верификацию. В первом случае, задача состоит в классификации речевого сигнала, при этом каждый класс соответствует одному человеку, зарегистрированному в системе. Во втором случае, задача представляет собой бинарную классификацию. Более формально, задача верификации представляет собой проверку статистической гипотезы, которую можно сформулировать следующим образом [11]. Предположим, что неизвестный произнес высказывание  $X$  и представляется пользователем  $S$ . Две противоположные гипотезы, в таком случае, это

$$\begin{cases} H_0 & : \text{ высказывание } X \text{ произнес } S \\ H_1 & : \text{ высказывание } X \text{ произнес не } S, \end{cases} \quad (1.1)$$

и ядро комплекса верификации должно определить, какую из двух гипотез необходимо принять.

Работу программного комплекса верификации источника речи можно разделить на четыре логические стадии:

- а) нормализация входного речевого сигнала;
- б) выделение характерных признаков;
- в) построение модели источника речи (стадия обучения);
- г) принятие по построенной модели и новой входной последовательности одной из двух гипотез (1.1).

Среди основных характерных особенностей в данной предметной области можно выделить следующие (по [10]):

- эффективность современных систем распознавания источника речи уменьшается, если входной сигнал зашумлен, канала передачи и прочих внешних причин. Важным моментом является попытка улучшить сигнал на этой предварительной стадии [9];
- дискриминативная точность моделей теряется при появлении в речи девиантных составляющих, привносимых различными психологическими состояниями говорящего (эмоции, напряжение). Существуют решения, позволяющие компенсировать данные составляющие [5];

- для биометрической аутентификации человека современным системам требуется значительное количество речевых данных (порядка десятков минут на человека). В таких условиях возрастает и требовательность системы к вычислительным ресурсам. Применимость технологий верификации по голосу увеличится, если удастся разработать технологии, позволяющие получить удовлетворительную производительность с использованием относительно малого количества данных (10-15 секунд). Существующим решением является использование модели гауссовой смеси с использованием так называемой универсальной фоновой модели (UBM – *Universal Background Model*) [16].

### 1.2.2 Нормализация входного речевого сигнала

Важной стадией голосовой аутентификации является предварительная подготовка речевого сигнала до выделения характерных признаков. Основные применяемые в данной области методы нормализации:

- удаление из записи фрагментов, не содержащих речь («тишины»). Самым простым и прямолинейным способом является определение уровня «энергии сигнала» и удаление тех кадров, в которых мощность не превышает заданный уровень, соответствующий внешнему шуму. Другой широко распространенный метод основан на подсчете количества изменений знака амплитуды внутри кадра (*zero-crossing rate*). Предложены также комбинации данных методов [9], [18];
- подавление шума, вызванного окружением. В данном случае используются методы нормализации, такие как нормализация по средним (*mean normalization*). Метод заключается в нахождении среднего значения энергии кадра сигнала и вычитании его из всех сэмплов (чаще данный метод используется после выделения характерных признаков);
- частотная развертка. В данном случае, используя частотный спектр сигнала, фильтруются частоты, находящиеся вне заданного диапазона, обычно [300Гц, 8000Гц], соответствующего спектру речи человека.



В данной работе применены все три метода нормализации, так как они независимы друг от друга и, применяемые совместно, позволяют получить точность выше, чем по отдельности [10].

### 1.2.3 Выделение характерных признаков

После нормализации сигнала, необходимо выделить признаки, которые характеризуют особенности вокального тракта человека. В сфере обработки сигналов и, в частности, распознавания речи и верификации по голосу, наиболее активное применение нашли так называемые мел-частотные коэффициенты кепстра (*MFCC*), отражающие психофизическое восприятие звука человеком. Существуют также исследования, в которых предлагается использование *LPCC* (*Linear prediction coding coefficients*), вейвлет-преобразований [22], а также различных их комбинаций. Однако, отдельные исследования показали [19], что LPC коэффициенты деградируют, если сигнал зашумлен. В данной работе использованы характерные признаки, основанные на MFCC, как наиболее изученные и подтвердившие свою эффективность в ряде исследований [10].

Речевой сигнал изначально представляет собой массив значений амплитуд, полученный с помощью дискретизации исходного аналогового сигнала с некоторой частотой сэмплирования  $f_s$ . На рисунке 1.1 представлен пример такого входного сигнала.

Для получения мел-частотных коэффициентов кепстра из входного сигнала, необходимо произвести следующие действия:

- а) исходный сигнал разбивается на кадры фиксированного размера (10-30мс) с использованием окон Хэмминга;
- б) для каждого кадра выполняется дискретное преобразование Фурье;
- в) полученные коэффициенты возводятся в квадрат и отображаются в мел-пространство с помощью перекрывающихся треугольных окон (*mel-scale filterbank*);
- г) для полученного массива выполняется дискретное преобразование косинусов (*Discrete Cosine Transform*), массив в данном случае рассматривается как сигнал;

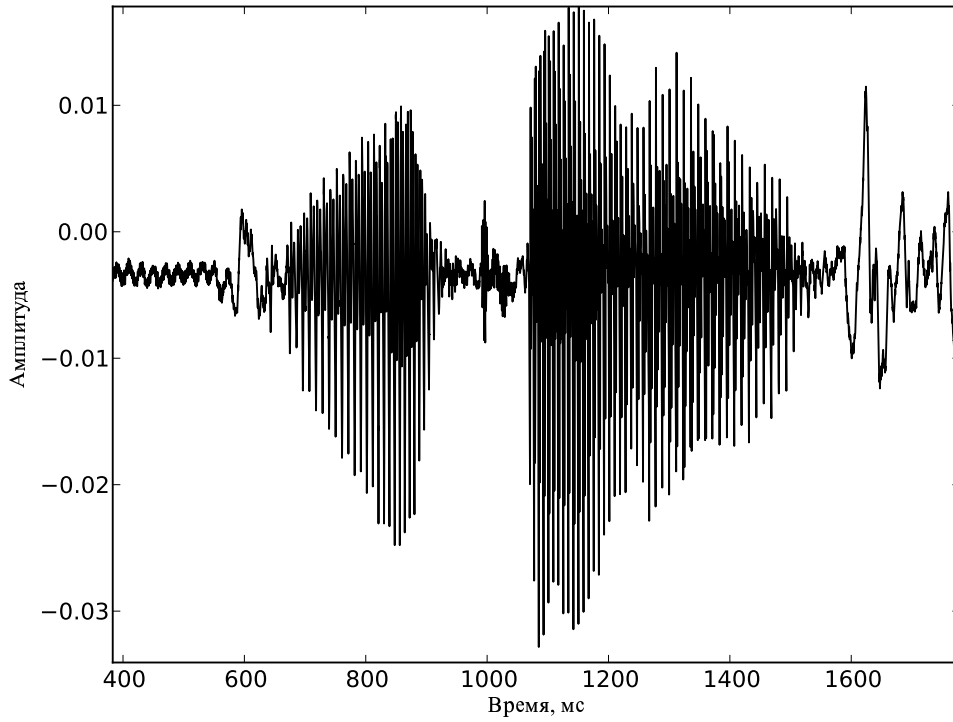


Рисунок 1.1 — Пример речевого сигнала (запись слова «Николай»)

- д) коэффициенты преобразования и являются мел-частотными коэффициентами кепстра.

В итоге для каждого кадра из исходного речевого сигнала получаем  $N$  коэффициентов  $\vec{M} = (M_1, M_2, \dots, M_N)$ . Первый коэффициент  $M_1$  исключается из вектора признаков, так как в действительности представляет собой «энергию» сигнала [15]. В дополнение к ним, для уменьшения влияния канала передачи сигнала, в современных системах используют [15] разницу коэффициентов, находящихся на расстоянии  $W$  кадров от текущего ( $t$ ):

$$\Delta \vec{M} = \vec{M}_{t+W} - \vec{M}_{t-W}. \quad (1.2)$$

Последние также называют *динамическими* характеристиками сигнала, тогда как мел-частотные коэффициенты являются *статическими*.

#### 1.2.4 Построение модели источника речи

Целью данного этапа является построение модели источника речи по специфичным для него векторам характеристических признаков.

В ранних исследованиях в области определения источника речи использовалось прямое сопоставление шаблонов. В сопоставлении шаблонов тестируемый

вектор и вектор-образец (обучающий вектор) сравниваются напрямую с помощью некоторой меры похожести, вычисление которой является ключевым при данном подходе. Изначально были использованы Евклидово расстояние или расстояние Махаланобиса. В конце 70-х японские ученые предложили [17] концепцию динамической деформации времени (*Dynamic Time Warping, DTW*), изначально для распознавания слов. Впоследствии, Фуруи [8] предложил использовать данный алгоритм для тексто-зависимой идентификации источника речи. Основным недостатком данного метода считается его высокая требовательность к вычислительным ресурсам по мере увеличения числа характеристических векторов. Однако, в последнее время предложены методы [12], позволяющие с помощью эффективного вычисления нижних границ меры уменьшить вычислительную нагрузку алгоритма *DTW*.

Следующий шаг в исследовании данной области был сделан в направлении алгоритмов кластеризации. Алгоритм К-средних был использован для получения специфичных для источника речи кодовых векторов для векторного квантования [20]. Был также предложен подход на основе теории нечетких множеств (нечеткое векторное квантование с использованием алгоритма С-средних) [1], показавший лучшие результаты, по сравнению со стандартным алгоритмом К-средних.

Аппарат скрытых Марковских моделей (*Hidden Markov Models, HMM*), успешно применяемый для распознавания речи, был также использован для задачи голосовой идентификации/верификации [13], [6]. Основным предположением является то, что текущее состояние модели зависит только от предыдущего. В контексте скрытых Марковских моделей, решение задачи верификации требует построения моделей фонов, составляющих речевой сигнал.

В 1995 Рейнольдс предложил модели смесей Гауссиан для решения задачи идентификации/верификации по голосу [15]. Это наиболее широко распространенный подход, основанный на аппарате математической статистики. Цель данного метода – моделирование функции плотности распределения векторов характерных признаков, в общем случае нелинейной. В данном методе, сложное распределение моделируется с помощью взвешенной суммы плотностей многомерных нормальных распределений (компонент смеси).

В качестве примера, на рисунке 1.2 показана гистограмма (выборочная плотность) распределения отдельно взятого кепстрального коэффициента для за-

писи речи диктора. Речь была записана в профессиональной студии, длительность записи после удаления тишины – 126 секунд. На рисунке 1.3 показана плотность распределения полученной по данным записи модели.

Размерность каждого из распределений  $D$  совпадает с размерностью вектора характерных признаков. Более формально, плотность распределения описывается следующим выражением:

$$p(\vec{x}|\lambda) = \sum_{i=1}^K \omega_i p_i(\vec{x}), \quad (1.3)$$

где  $K$  – количество компонент,

$\vec{x}$  – вектор характерных признаков,

$\omega_i$  – вес  $i$ -ой компоненты, веса удовлетворяют условию  $\sum_{i=1}^K \omega_i = 1$ ,

$p_i$  – плотность распределения  $i$ -ой компоненты.

Плотность распределения каждой компоненты – это  $D$ -мерный Гауссиан [23]:

$$p_i(\vec{x}) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} e^{-1/2(\vec{x}-\vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}-\vec{\mu}_i)}, \quad (1.4)$$

где  $\vec{\mu}_i$  – вектор математического ожидания,

$\Sigma_i$  – матрица ковариаций.

Плотность распределения смеси Гауссиан полностью описывается параметрами компонент и значениями весов. Параметры модели представим в следующей нотации:

$$\lambda = \{\omega_i, \vec{\mu}_i, \Sigma_i\}, i = \overline{1, K}. \quad (1.5)$$

Каждый пользователь представлен собственной моделью  $\lambda$ .

Модель смеси Гауссиан может иметь различные типы, в зависимости от вида матриц ковариаций  $\Sigma_i$ . В одном случае может использоваться диагональная матрица (все элементы, кроме диагональных, всегда равны нулю, диагональные элементы в данном случае являются значениями средне-квадратичных отклонений  $\sigma^2$ ). В другом случае используются полные матрицы ковариаций. Также возможно использование общей, глобальной матрицы для всех компонент. Эксперименты, проведенные в рамках основополагающей работы по данной теме [15], показали, что лучший результат может быть получен с использованием диагональных

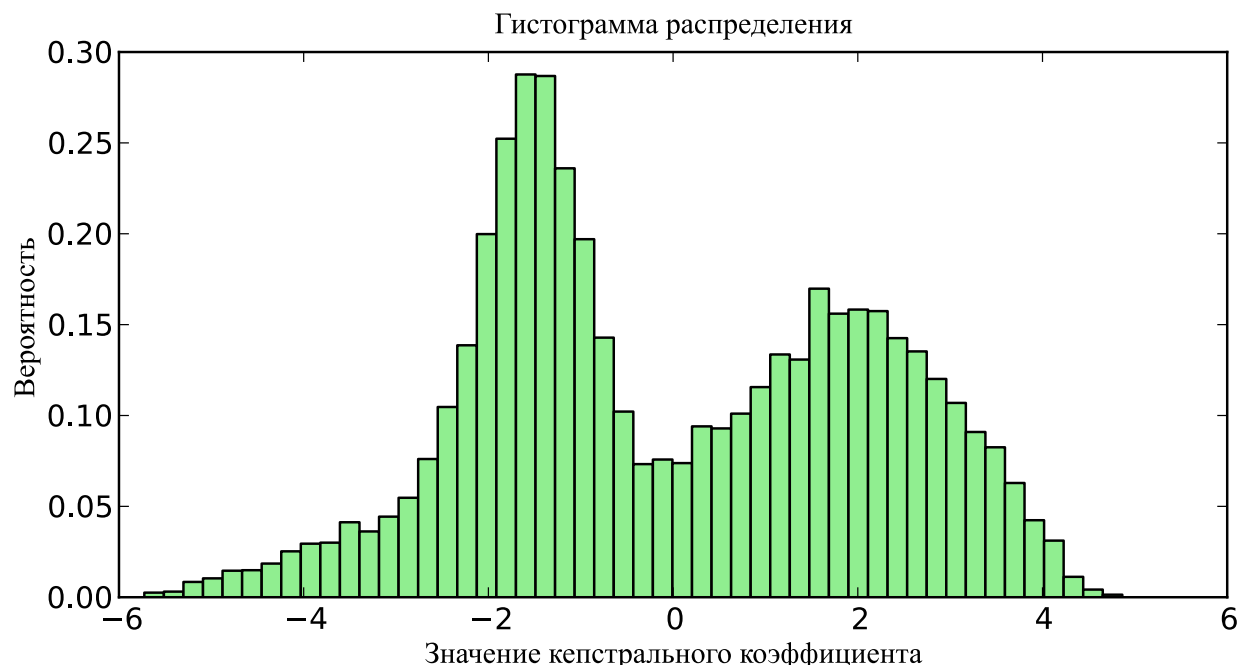


Рисунок 1.2 — Пример выборочной плотности распределения кепстрального коэффициента

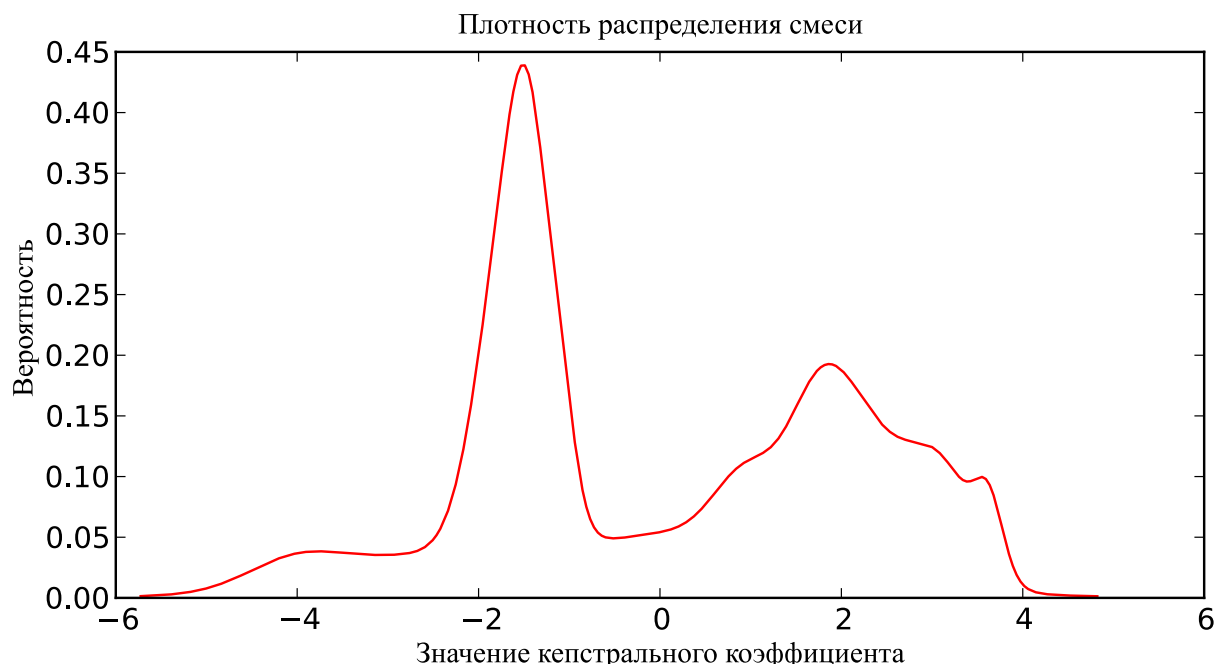


Рисунок 1.3 — Аппроксимация функции плотности распределения кепстрального коэффициента с помощью смеси Гауссиан

матриц ковариаций, отдельных для каждой из компонент. Основным аргументом против использования диагональных матриц является тот факт, что диагональ-

ные матрицы подразумевают статистическую независимость коэффициентов вектора характерных признаков. Однако, как показали эксперименты Рейнольдса, смоделировать эту зависимость можно путем введения дополнительных компонент в смесь. Так как вычисления в случае диагональных матриц значительно упрощаются (нет необходимости инвертировать матрицу ковариаций при вычислении (1.4)), автор предлагает использование именно этого варианта.

Для определения параметров модели  $\lambda$  существуют несколько методов, наиболее распространенным из которых является метод максимального правдоподобия [21]. Задача метода состоит в нахождении по заданным обучающим данным таких параметров модели, при которых функция правдоподобия модели достигает максимума. Для последовательности из  $T$  обучающих векторов  $X = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_T\}$ , функция правдоподобия может быть записана как

$$p(X|\lambda) = \prod_{t=1}^T p(\vec{x}_t|\lambda). \quad (1.6)$$

Прямая максимизация выражения (1.6) невозможна, так как функция от параметров  $\lambda$  нелинейна. Однако, приближенные значения могут быть получены с помощью алгоритма ЕМ (*Expectation-Maximization*) [4].

После этапа обучения, мы имеем готовую модель  $\lambda$ , соответствующую данной персоне. Теперь, по данной тестовой последовательности, мы можем вычислить функцию правдоподобия  $p(S|H_0)$  из выражения (1.1).

Основным вопросом остается вычисление второй функции,  $p(S|H_1)$ . По определению, она равна вероятности того, что тестовая последовательность произнесена кем-то, кроме  $S$ . В научном сообществе существуют два основных подхода к моделированию альтернативной гипотезы: универсальная фоновая модель («мировая» модель) и когортные модели.

Универсальная фоновая модель (*Universal Background Model, UBM*) – это модель, цель которой характеризовать всех возможных источников речи «мира» во всех возможных контекстах. Данная модель обучается на большом количестве (несколько часов) речевых данных от множества источников речи, сбалансированного по гендерному типу, а также, по возможности, по оборудованию и стандарт-

ным условиям. Вычисление правдоподобия  $H_1$  в таком случае осуществляется аналогично  $H_0$ :

$$p(S|H_1) = p(X|\lambda_{UBM}), \quad (1.7)$$

где  $\lambda_{UBM}$  – универсальная фоновая модель.

Когортные модели (*Cohort speaker models*) – это способ оценки правдоподобия гипотезы  $H_1$ , при котором вместе моделирования всего мира, из имеющихся моделей выделяется небольшое подмножество, называемое «когортным множеством»,  $\mathbb{C}_S$ . Возникают следующие задачи:

- необходимо выбрать способ выделения когортного множества  $\mathbb{C}_S$ . Существуют два противоположных по смыслу подхода: определять по мере правдоподобия наиболее близкие к целевой модели на стадии обучения; выбирать наиболее близких по мере правдоподобия к тестируемому образцу (на стадии верификации). В [11] показано, что второй способ позволяет добиться меньшего показателя ошибки;
- необходимо выбрать способ получения меры правдоподобия гипотеза  $H_1$  на основе мер моделей из когортного множества. Широко распространены два метода: взятие максимума из значений и взятие среднего арифметического.

Очевидным недостатком данного подхода является его вычислительная нагрузка по сравнению с UBM. Вместо того, чтобы вычислять  $p(S|H_1)$  по выражению (1.6), необходимо выбирать  $N_{\mathbb{C}_S} = |\mathbb{C}_S|$  когортных моделей из базы, что приводит к вычислению аналогичного выражения для *всех* моделей, отличных от целевой. В [11] предложен метод предварительного квантования моделей, при котором значительно уменьшается круг поиска когортных моделей.

Подход на основе смеси Гауссиан показывает результат для задачи идентификации на уровне 94.5% лишь в случае большого количества данных на стадии обучения (60-90 секунд для источника речи) [15]. Поэтому были предложены различные методы адаптации, основным из которых можно выделить Байесово обучение [16]. В данном методе модель источника речи получают путем адаптации параметров «мировой» модели по обучающим векторам данного человека.

В данной работе используется подход на основе смеси Гауссиан как наиболее развитый и теоретически обоснованный метод получения модели источника речи для голосовой верификации.

### 1.2.5 Принятие решения

Предположим, что меры правдоподобия обеих гипотез (1.1) известны. Рассмотрим *отношение правдоподобия*:

$$LR_{H_0, H_1} = \frac{p(S|H_0)}{p(S|H_1)}, \quad (1.8)$$

в таком случае, отношение правдоподобия дает оптимальное решение в Байесовом смысле (классификация с минимизацией рисков) [7]. Тогда, процедура принятия решения выглядит следующим образом:

$$\text{Решение} = \begin{cases} H_0, & LR_{H_0, H_1} > \Theta_S \\ H_1, & LR_{H_0, H_1} \leq \Theta_S \end{cases} \quad (1.9)$$

Ключевым моментом в (1.9) является выбор числа  $\Theta_S$  (величины порога для пользователя  $S$ ). Величина порога  $\Theta_i$  определяется из образцов для обучения таким образом, чтобы обеспечивался необходимый уровень ошибок 1-ого и 2-ого рода. Порог также может быть глобальным для всех пользователей. При использовании отдельного порога для каждого из источников речи возникает проблема нахождения величины этого порога, так как это величина, определяемая экспериментально. Для этого необходимо иметь дополнительные записи фраз для тестирования полученной модели и определения ее чувствительности по отношению к данному источнику речи. Данное требование осложняет процесс регистрации в системе, поэтому следует предусмотреть как персональный, так и глобальный порог вхождения.

## 1.3 Выводы

В результате проведенного исследования предметной области, можно сформулировать следующие выводы:

- системы голосовой аутентификации в данный момент не имеют широкого применения в веб-среде;



- на процесс голосовой аутентификации влияют внешние шумы, поэтому необходимо предусмотреть методы нормализации (описанные в разделе ??);
- подход на основе использования смеси Гауссиан является наиболее изученным и теоретически обоснованным, поэтому он должен быть использован при разработке программного комплекса;
- в связи со спецификой метода (тексто-независимостью), необходимо разработать алгоритм фильтрации входной последовательности для удаления сегментов, не содержащих речи (удаление тишины);
- необходимо получить параметры для модели, которая будет использована как универсальная фоновая модель (см. раздел 1.2.4);
- необходимо определить экспериментально величину порога вхождения для использования в разрабатываемом программном комплексе;

## 2 Конструкторский раздел

Данный раздел содержит описание архитектуры разрабатываемого программного комплекса, описание подсистем, из которых состоит комплекс, а также описание спроектированного алгоритма нормализации входного сигнала.

### 2.1 Общая архитектура разрабатываемого комплекса

Общая архитектура комплекса представлена на рисунке 2.1.

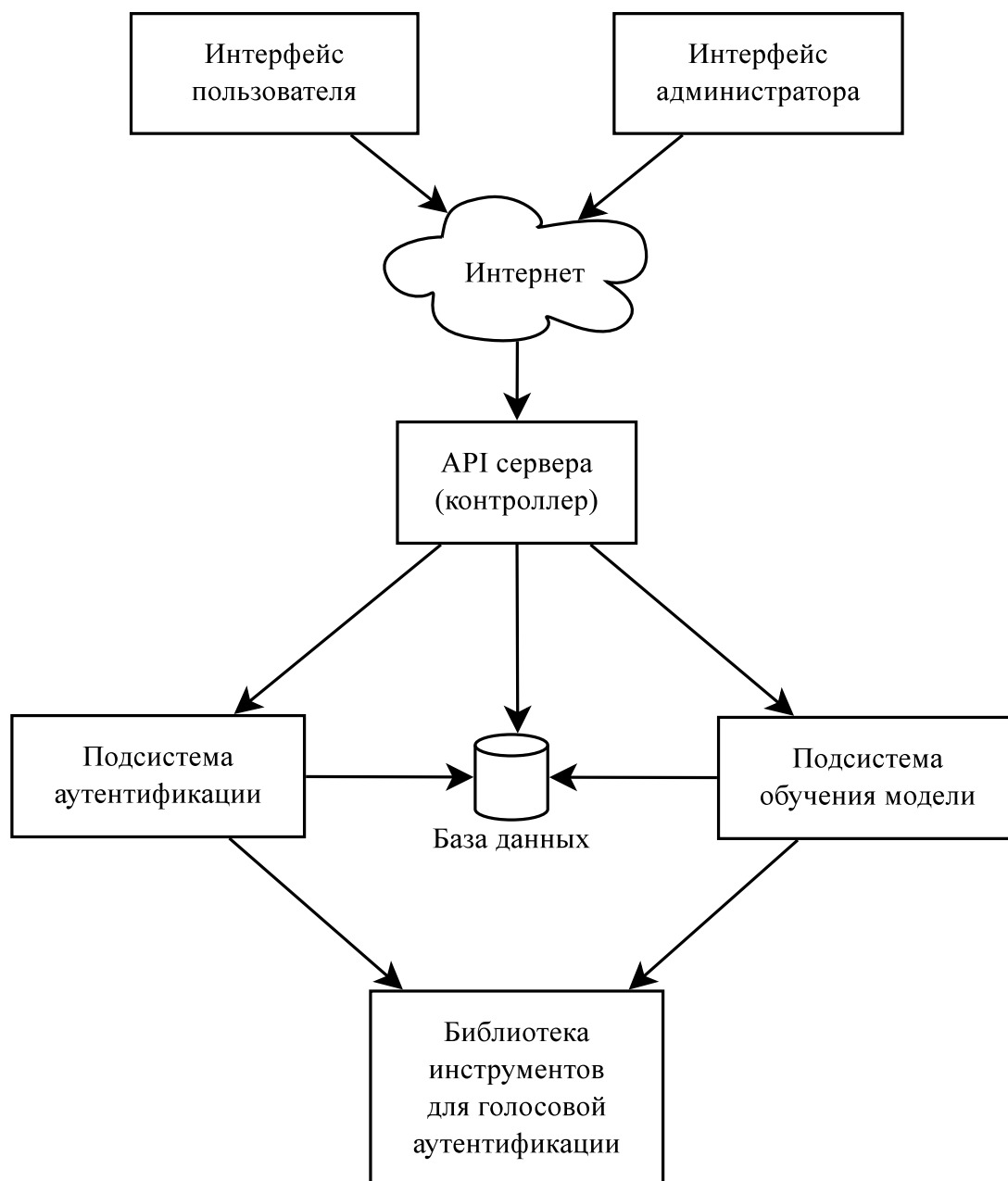


Рисунок 2.1 — Общая архитектура разрабатываемого программного комплекса

В разрабатываемом программном комплексе можно выделить четыре основные подсистемы, перечисленные по уровню абстракции:

- интерфейс пользователя – приложение, позволяющее пользователю производить регистрацию и аутентификацию в программном комплексе, реализующее передачу данных с микрофона пользователя на сервер (для сохранения в базе данных и последующей обработки);
- контроллер сервера – часть комплекса, выполняющая роль посредника между запросами пользователя и остальными подсистемами. Задача контроллера состоит в обработке запросов от интерфейса пользователя, передаче управления в соответствующие модули программного комплекса, обработке и выдаче результатов обратно в интерфейс;
- подсистема обучения модели (подсистема регистрации) – позволяет получить по заданным речевым последовательностям (набору звуковых файлов в базе) персональную модель пользователя, сохранив ее в базе для последующего использования в процессе аутентификации;
- подсистема голосовой аутентификации – подсистема, позволяющая определить, принадлежит ли заданная речевая последовательность (в виде набора звуковых файлов в базе) заданному пользователю (персональная модель которого создается на этапе регистрации);
- подсистема хранения речевых данных пользователей (база данных);
- библиотека инструментов для голосовой аутентификации – ядро программного комплекса, реализующее подходы, выбранные в результате обзора существующих решений в разделе 1.2 и предоставляющее интерфейс для подсистем голосовой аутентификации и регистрации.

## **2.2 Варианты использования**

На рисунке 2.2 представлена диаграмма вариантов использования программного.

Пользователей программного комплекса можно разделить на две категории:

- пользователь программного комплекса голосовой аутентификации – основной пользователь, использующий комплекс для прохождения аутентификации на некотором ресурсе. Пользователь имеет возможность зарегистрироваться в программном комплексе, получив при этом персональную мо-

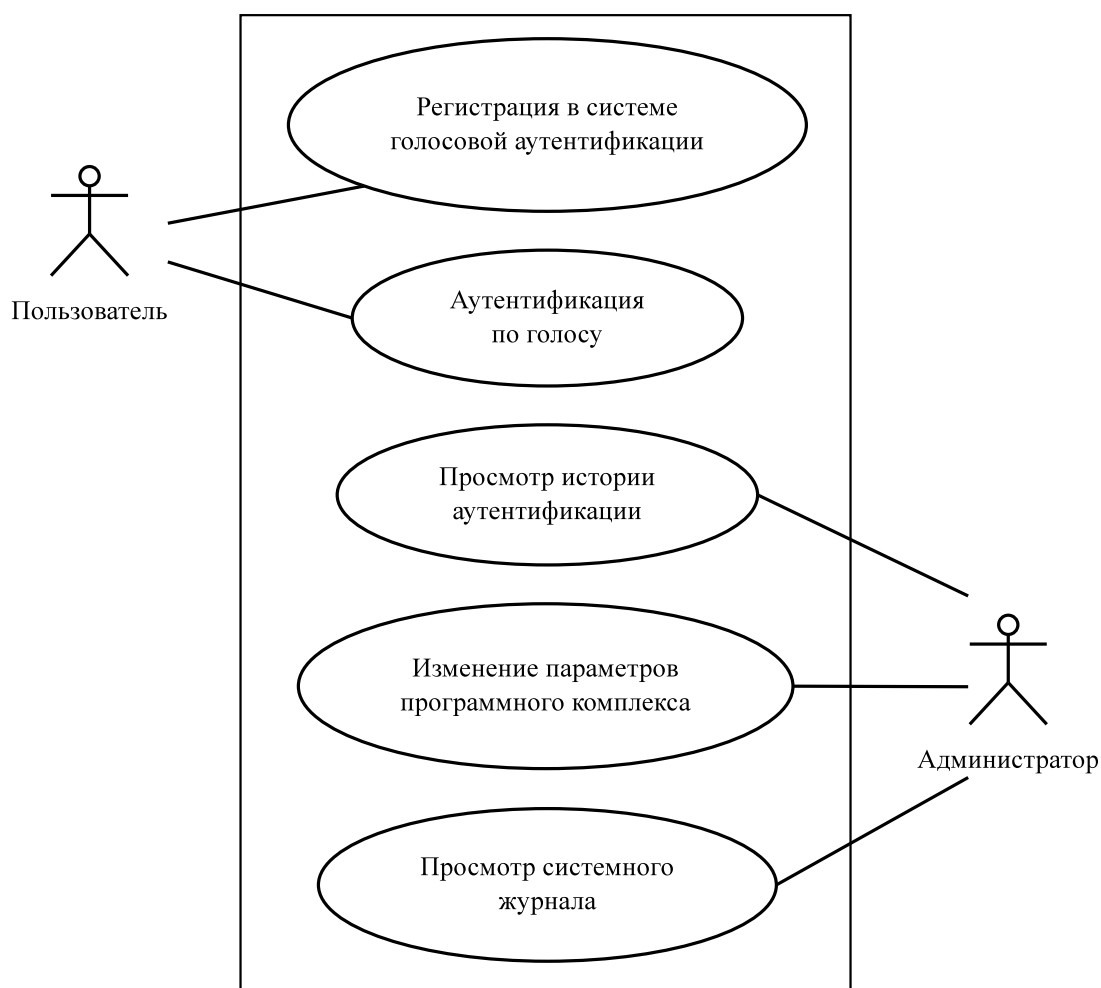


Рисунок 2.2 — Диаграмма вариантов использования программного комплекса

дель источника речи, обученную на нескольких образцах фразы, которую он должен повторить в процессе регистрации. После прохождения процесса регистрации пользователь может использовать программный комплекс для аутентификации на сайте, при этом, пройдя аутентификацию, пользователь будет обладать теми же правами, которыми он обладал бы, войдя на ресурс традиционным способом (по паролю);

- администратор – пользователь, наделенный правами на просмотр истории посещений через специализированный интерфейс. Помимо этого, данный тип пользователя может добавлять, редактировать и удалять параметры, влияющие на процесс аутентификации, такие как пути к файлам моделей источников речи, текущая активная модель источника речи для пользователя, величина входного порога для пользователя (определяющий параметр для принятия решения, описанный в разделе 1.2.5). Помимо этого, администратор имеет возможность просматривать журнал, в котором ведется

запись основных событий приложения, а также фиксируются возникающие ошибки и предупреждения.

### 2.3 Подсистема обучения модели (стадия регистрации)

При обучении модели пользователю необходимо предоставить несколько образцов, записей «кодовой фразы», ту же фразу он должен будет впоследствии произнести для входа на защищаемый ресурс по голосу. Для обзора прохождения процесса регистрации с точки зрения времени, рассмотрим диаграмму последовательности, представленную на рисунке 2.3:

- а) Пользователь инициирует процесс регистрации, интерфейс пользователя посылает сообщение `register()` контроллеру;
- б) Контроллер на стороне сервера создает новую сессию записи, генерируя при этом уникальный код сессии (`session_id`), который передается клиенту и используется в дальнейшем для проверки аутентичности клиента. Состояние созданной сессии устанавливается в «Ожидание речевых данных»;
- в) Пользователь производит запись кодовой фразы несколько раз<sup>1</sup>, при этом каждая запись отправляется на сервер и сохраняется в базе, путь к записи добавляется в данные сессии;
- г) После завершения записи фразы, пользователь подтверждает регистрацию, при этом интерфейс отправляет контроллеру сообщение `confirm()`, после чего процесс регистрации переходит в состояние «Обучение модели» (`started`), а контроллер ставит соответствующую сессию записи в очередь на обучение, посылая сообщение `enroll` ядру программного комплекса (подсистеме регистрации);
- д) Интерфейс клиента опрашивает контроллер с заданной периодичностью о состоянии процесса регистрации. Опрос продолжается до тех пор, пока состояние не окажется одним из тупиковых: «Обучение завершено», «Ошибка при обучении» или «Обучение прервано». На диаграмме показана стереотипная ситуация, при которой регистрация происходит в штатном режиме (обучение проходит успешно).

---

<sup>1</sup>Необходимое количество должно быть определено экспериментально, см. раздел 4.

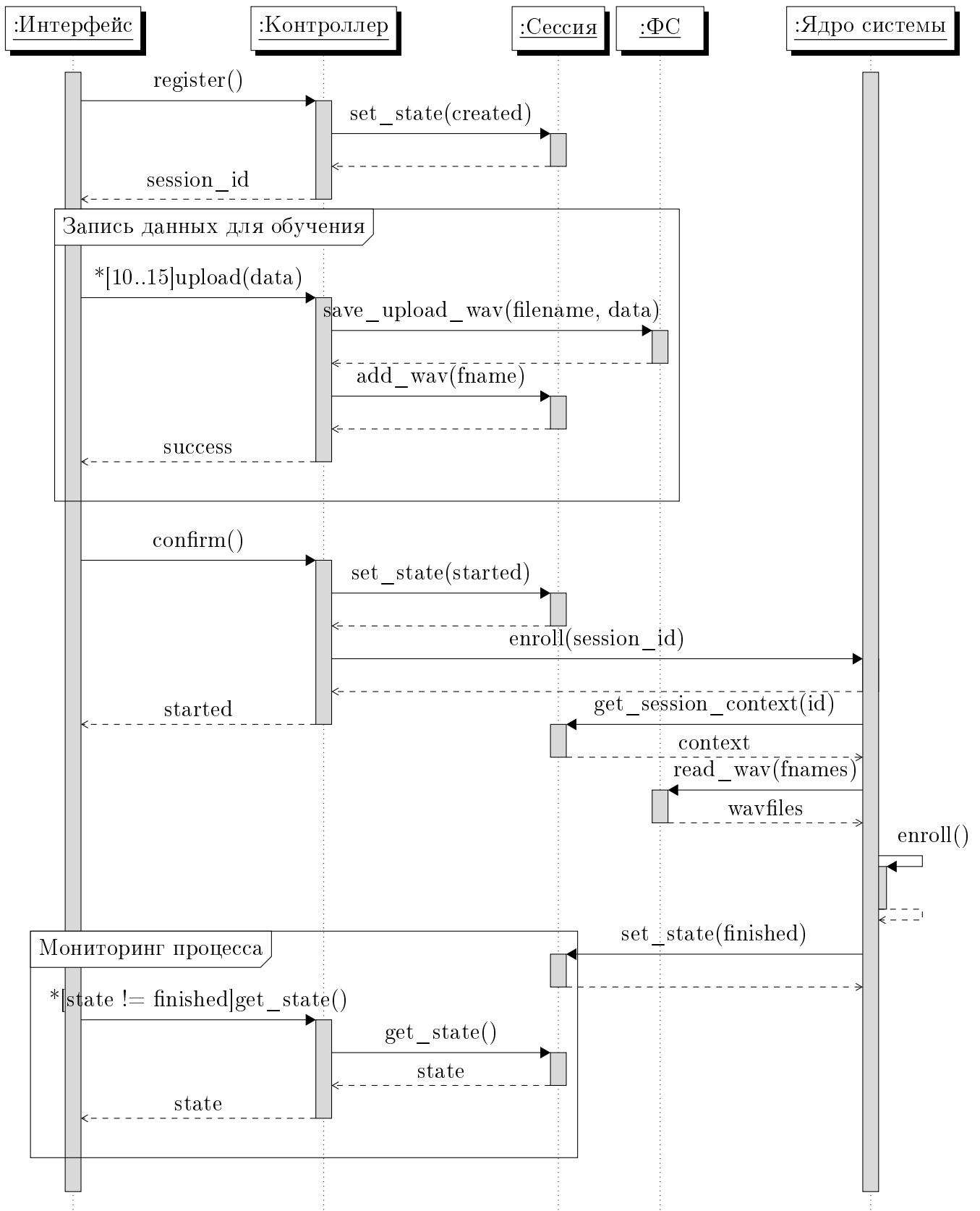


Рисунок 2.3 — Диаграмма последовательности: процесс регистрации в системе

е) Подсистема регистрации, обслуживающая очередь запросов на обучение модели от контроллера, получает новую заявку и начинает процесс обучения:

- 1) Получает контекст сессии (`get_session_context`), в котором содержатся пути к файлам записей, код сессии и другая информация;
- 2) Обращаясь к хранилищу (файловой системе), считывает в память файлы записей (`read_wav`);
- 3) Вызывает функцию библиотеки инструментов для аутентификации `enroll`, передавая параметры обучения, в том числе считанные данные.

Как видно из рисунка 2.3 и данного выше описания, сам процесс обучения происходит отдельно от взаимодействия клиентов и контроллера, поскольку обучение требует больших вычислительных ресурсов и не может происходить в том же потоке выполнения, что и клиент-серверное взаимодействие. В таком случае, процессы обучения рассматриваются подсистемой регистрации как заявки на обучения и ставятся контроллером в очередь, выполняясь по одиночке и, тем самым, снижая нагрузку на сервер.

На рисунке 2.4 представлена диаграмма состояний для процесса регистрации пользователя в программном комплексе голосовой аутентификации (то есть процесса обучения новой модели по речевым данным пользователя) с точки зрения сервера. Переходы между состояниями описаны выше при рассмотрении диаграммы последовательности.

На рисунке 2.5 представлена диаграмма состояний для процесса регистрации пользователя в программном комплексе голосовой аутентификации со стороны пользовательского интерфейса.

Опишем возможные состояния процесса регистрации с точки зрения интерфейса пользователя:

- создан — начальное состояние, ожидание начала записи кодовых фраз;
- запись — запись в процессе;
- останавливается — пользователь инициировал остановку записи (или остановка инициирована автоматически при достижении максимальной длины записи);





- остановлено — запись завершена и в данный момент находится в памяти клиента. В данном состоянии проверяется, достаточно ли длины записи для отправки на сервер (запись не должна быть слишком короткой<sup>1</sup>);
- недостаточно для отправления — запись слишком короткая. При этом возможно возобновление записи (или отмена процесса);
- отправка на сервер — запись достаточной длины, инициируется отправка данных на сервер. Находясь в данном состоянии, пользовательский интерфейс ожидает передачи данных и положительного ответа сервера, оповещающего о приеме;
- ошибка при отправлении — данное состояние возникает, если при отправлении данных на сервер возникла ошибка (соединение оборвалось, сервер не смог прочитать данные и т.п.);
- отправлено — отправление звуковой записи успешно завершено, пользовательский интерфейс готов к записи новой фразы. Из этого состояния возможен переход вновь в состояние записи, отмена процесса регистрации или же подтверждение (после отправления на сервер минимально допустимого для регистрации количества записей);
- подтверждено — сервер ответил, оповестив о постановке в очередь запроса на обучение модели, переход в следующее состояние;
- обучение в процессе — в данном состоянии интерфейс периодически опрашивает сервер о состоянии процесса обучения (см. рисунок 2.4);
- недостаточно данных для обучения — обучение завершилось неудачей из-за недостаточного количества речевых данных. В данном случае пользователь может вернуться к записи фразы или отменить процесс регистрации;
- ошибка при обучении — при обучении произошла непредвиденная ошибка, в данном случае повторное обучение по тем же данным невозможно, пользователю предложено повторить процесс сначала;
- обучение завершено — обучение успешно завершено, комплекс готов производить аутентификацию данного пользователя по голосу;
- отменено — в данное состояние возможен переход, если пользователь отказывается от продолжения процесса регистрации на некотором его этапе.

---

<sup>1</sup>Минимальная допустимая длина записи должна определяться экспериментально.

## 2.4 Подсистема голосовой аутентификации

В результате завершения процесса регистрации в программном комплексе, на сервере сохраняется персональная модель для пользователя. При этом для данного пользователя фиксируется индивидуальный порог вхождения (который впоследствии может быть скорректирован вручную). Модель пользователя, порог вхождения и универсальная фоновая модель – достаточный контекст для проведения процедуры аутентификации (подробнее см. разделы 1.2.4, 1.2.5).

На рисунке 2.6 представлена диаграмма последовательности для процесса аутентификации.

Диаграмма аналогична соответствующей диаграмме для процесса регистрации, поэтому в дальнейшем описании ограничимся лишь принципиальными различиями между двумя процессами. Рассмотрим основные этапы процесса аутентификации:

- а) Пользователь инициирует процесс аутентификации, интерфейс пользователя посылает сообщение `verify()` контроллеру;
- б) Контроллер на стороне сервера создает новую сессию записи, генерируя при этом уникальный код сессии (`session_id`), состояние созданной сессии устанавливается в «Ожидание речевых данных»;
- в) Пользователь производит запись кодовой фразы единожды, после чего запись отправляется на сервер (`upload(data)`) и сохраняется в базе (`save_upload_wav(filename, data)`);
- г) После сохранения записанной фразы в базе, контроллер ставит запрос на аутентификацию в очередь, посылая сообщение `verificate(session_id)` подсистеме аутентификации, обслуживающей очередь запросов. Состояние процесса при этом устанавливается в «Аутентификация в процессе»;
- д) Интерфейс клиента опрашивает контроллер с заданной периодичностью о состоянии процесса аутентификации. Опрос продолжается до тех пор, пока состояние не окажется одним из тупиковых: «Ошибка в процессе аутентификации» или «Аутентификация успешно завершена». В последнем случае, контроллер возвращает результат аутентификации (булево значение), предварительно завершая все необходимые действия, по аутентификации и авторизации сессии пользователя в интернет-портале.

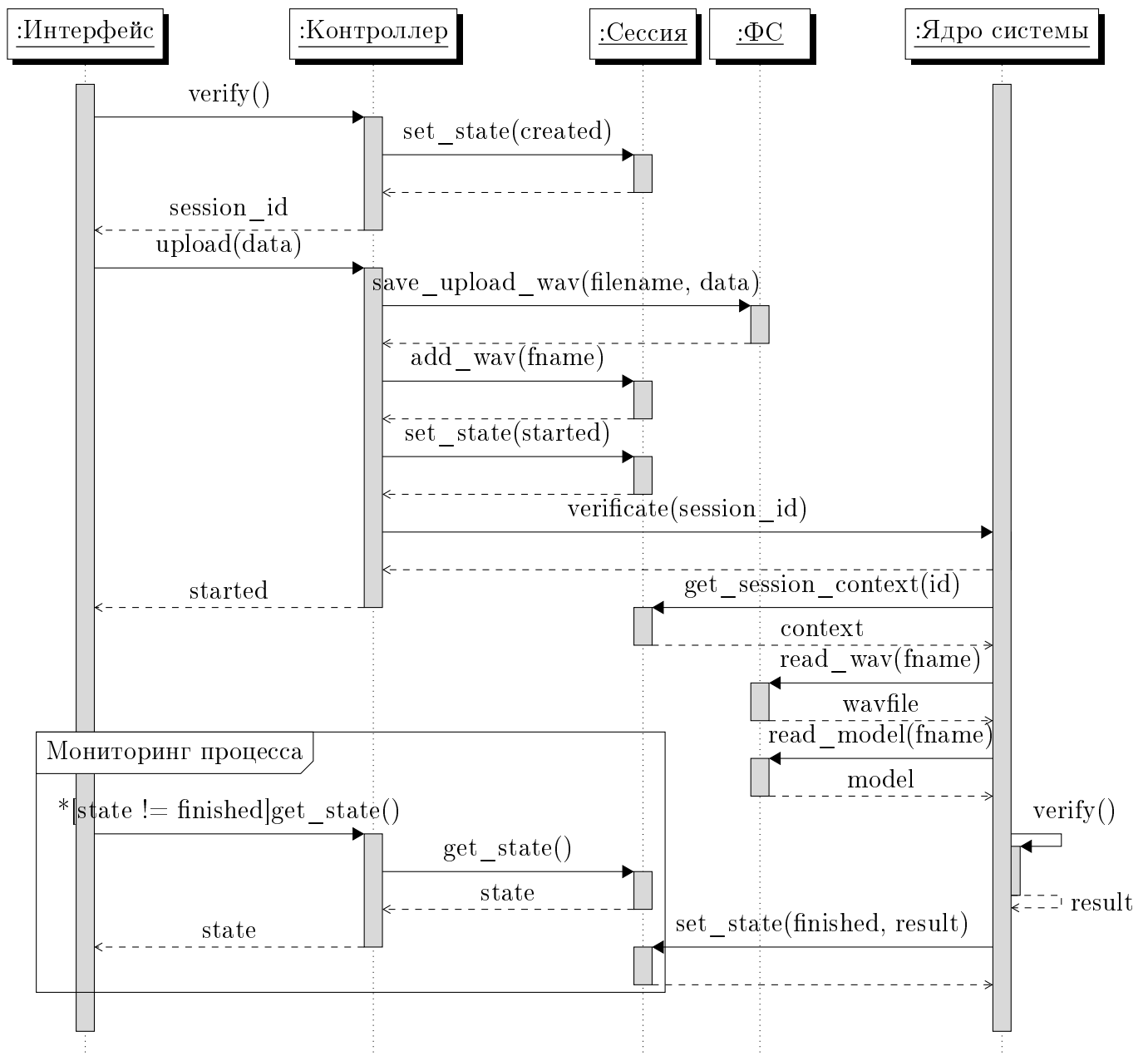


Рисунок 2.6 — Диаграмма последовательности: процесс аутентификации

е) Подсистема аутентификации, обслуживающая очередь запросов от контроллера, получает новую заявку и начинает процесс аутентификации:

- 1) Получает контекст сессии (`get_session_context`), в котором содержатся пути к файлам записей (обычно файл один), код сессии и другая информация;
- 2) Обращаясь к хранилищу (файловой системе), считывает в память файлы записей (`read_wav`), а также модель источника речи, созданную в процессе регистрации и универсальную фоновую модель (`read_model`);
- 3) Вызывает функцию библиотеки инструментов для аутентификации `verify`, передавая модель пользователя, универсальную фоновую модель, порог вхождения и считанные речевые данные.

На рисунке 2.7 представлена диаграмма состояний для процесса аутентификации пользователя по голосу с точки зрения сервера. Переходы между состояниями описаны выше при рассмотрении диаграммы последовательности.

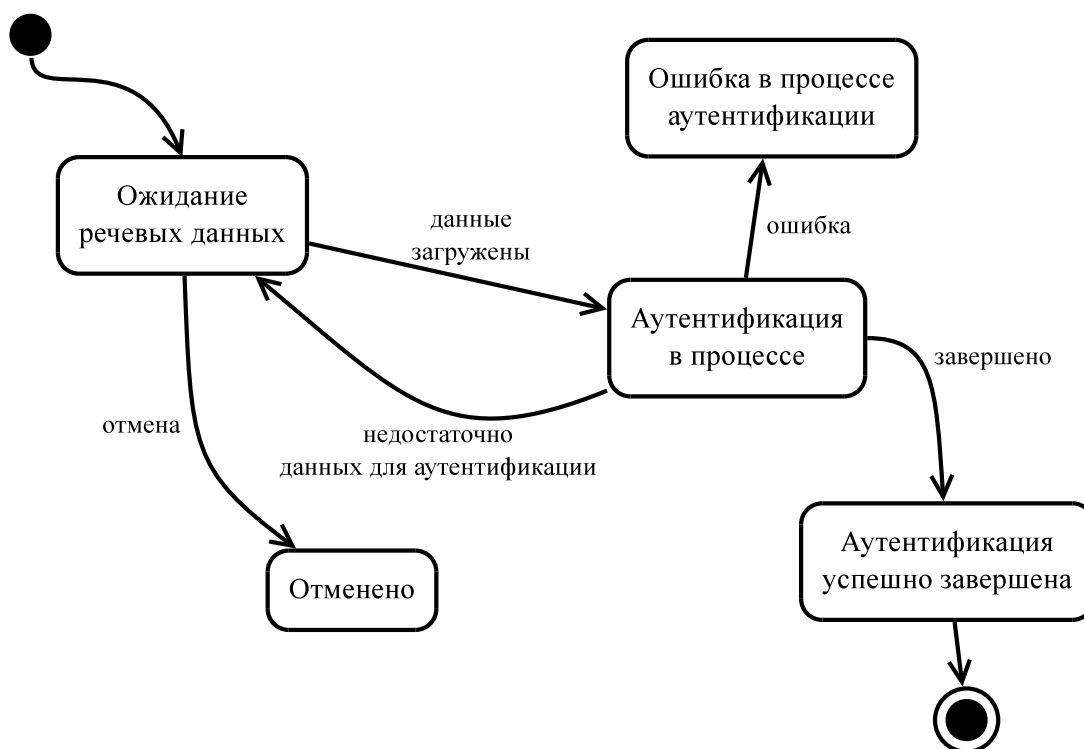


Рисунок 2.7 — Диаграмма состояний для процесса аутентификации (сервер)

На рисунке 2.8 представлена диаграмма состояний для процесса аутентификации пользователя по голосу с точки зрения пользовательского интерфейса.

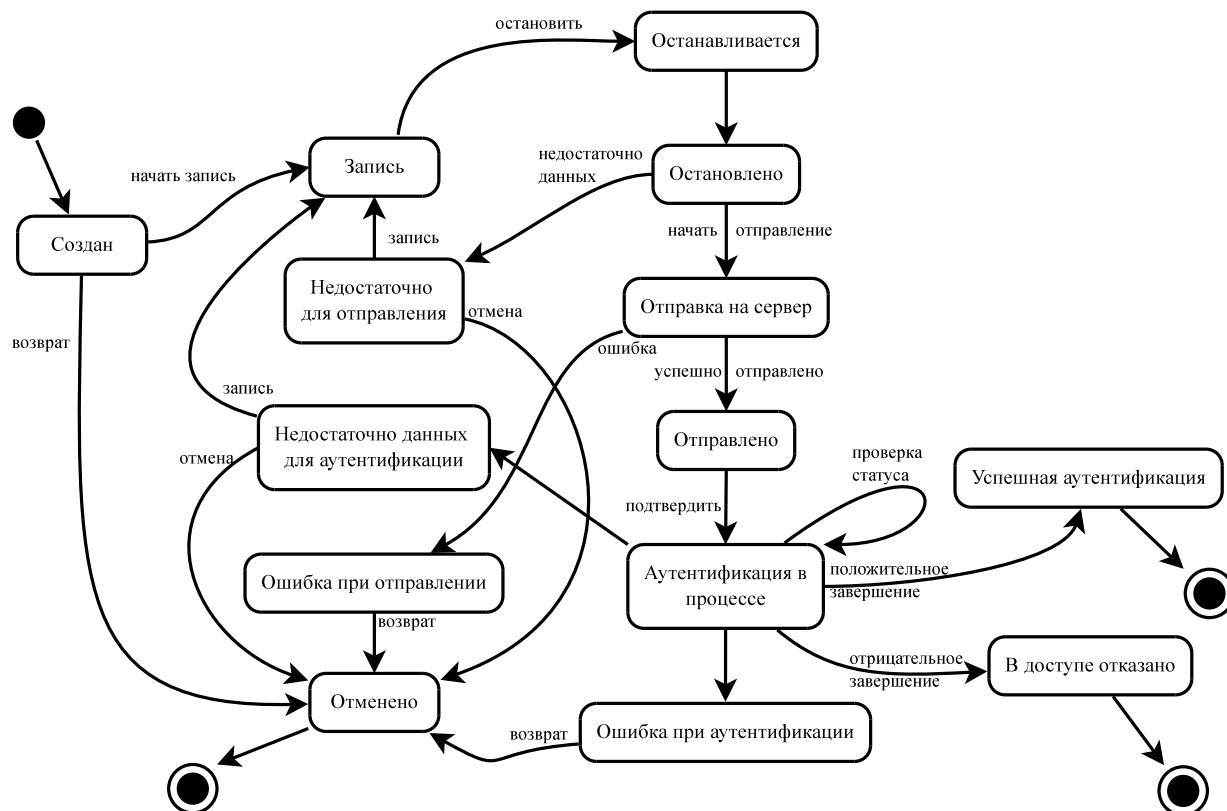


Рисунок 2.8 — Диаграмма состояний для процесса аутентификации (клиент)

Состояния и переходы аналогичны соответствующим для процесса регистрации (рисунок 2.5), опишем принципиальные различия:

- Конечное состояние в случае успешного завершения процесса аутентификации распадается на два: «Успешная аутентификация» (подразумевается результат аутентификации, а не успешное завершение самого процесса) и «В доступе отказано». В первом случае пользователь перенаправляется на ресурс, который он запросил (или на страницу по-умолчанию).
- После отправки записи на сервер, интерфейс пользователя сразу переходит в состояние «Аутентификация в процессе» (аналогичное состоянию «Обучение в процессе»), так как для аутентификации требуется только одна запись кодовой фразы.

## 2.5 Подсистема хранения голосовых данных

На рисунке 2.9 представлена диаграмма сущность-связь для разрабатываемой базы данных. Рассмотрим подробнее сущности, представленные на данной диаграмме:

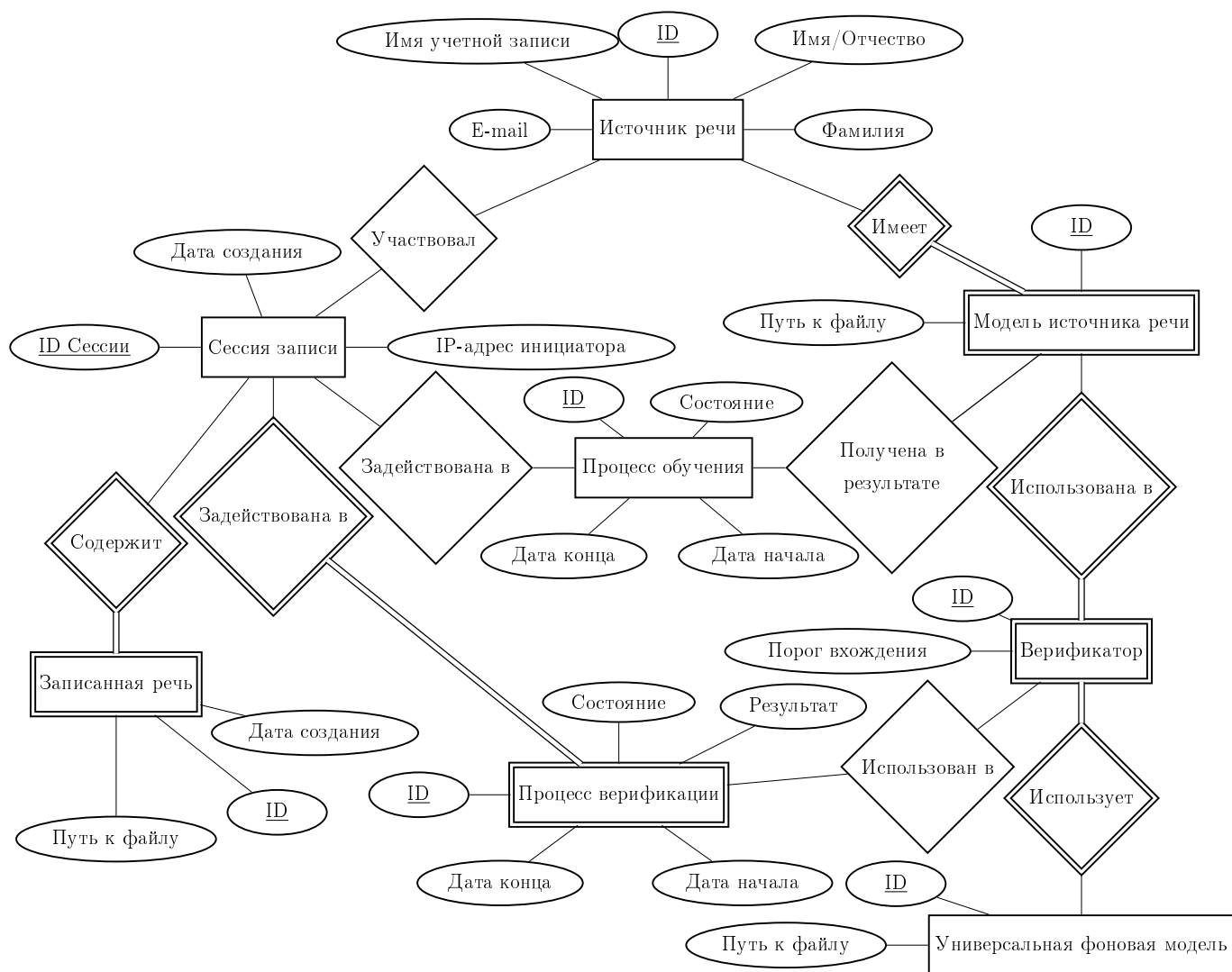


Рисунок 2.9 — Диаграмма сущность-связь подсистемы хранения голосовых данных

- **источник речи.** Пользователь программного комплекса голосовой аутентификации, имеющий уникальное *имя учетной записи*. Может инициировать **сессию записи**, которая, в свою очередь, может быть использована для процесса обучения персональной **модели источника речи**;
- **модель источника речи.** Персональная модель, параметры которой вычисляются на этапе регистрации, после чего сохраняются в файл для последующего использования, в базе хранится *путь к файлу*;
- **универсальная фоновая модель.** Глобальная модель, параметры которой вычисляются заранее, сохраняются в файл для последующего использования. Требуется для процесса аутентификации, в соответствии с описанием в разделе 1.2.4;

- **сессия записи.** Представляет собой сеанс работы пользователя, предполагающий запись голосовых данных. Сессия записи состоит из одной или нескольких записанных высказываний. Для защиты сессии записи используется уникальный идентификатор сессии, который генерируется на стороне сервера в момент создания сессии и передается клиенту для проверки его последующих запросов на аутентичность. Также записывается *IP-адрес пользователя*, с которого совершаются запросы;
- **верификатор.** Сущность, представляющая собой контекст, необходимый для выдачи решения об аутентификации по заданной речевой последовательности. Верификатор объединяет персональную **модель источника речи**, необходимую для получения меры правдоподобия основной гипотезы, **универсальную фоновую модель**, необходимую для получения меры правдоподобия альтернативной гипотезы, а также хранит значение *порога вхождения* для данной модели источника речи ( $\Theta_S$  из отношения 1.9);
- **записанная речь.** Фраза, записанная пользователем и переданная на сервер. При этом запись в виде звукового файла сохраняется в хранилище, а в базе хранится путь к этому файлу и дата добавления записи;
- **процесс обучения.** Сущность представляет собой процесс создания **модели источника речи**. Именно она хранит информацию о *состоянии* (см. раздел 2.3), а также о времени создания и завершения. Обучение производится на основе **сессий записи** (обычно на основе одной сессии);
- **процесс аутентификации.** Сущность представляет собой процесс аутентификации. Объект аутентификации — **сессия записи**. Помимо аналогичных атрибутов *Состояние*, *Дата начала* и *Дата конца*, также имеется атрибут *Результат*, который содержит булево значение, отражающее факт аутентичности.

## 2.6 Ядро комплекса голосовой аутентификации

Как было указано в разделе 2.1, библиотека инструментов для аутентификации является ядром программного комплекса и расположена на нижнем уровне в его архитектуре. Библиотека представляет собой набор расширяемых классов и

функций для полного цикла обработки сигнала (от считывания звукового файла в память и разбора формата до получения массива характерных признаков, готового для обучения или верификации), получения параметров модели источника речи по массиву характерных признаков (обучение модели), а также для собственно верификации по заданной модели и новому массиву характерных признаков.

На рисунке 2.6 показана диаграмма наследования для классов, составляющих библиотеку инструментов для голосовой аутентификации.

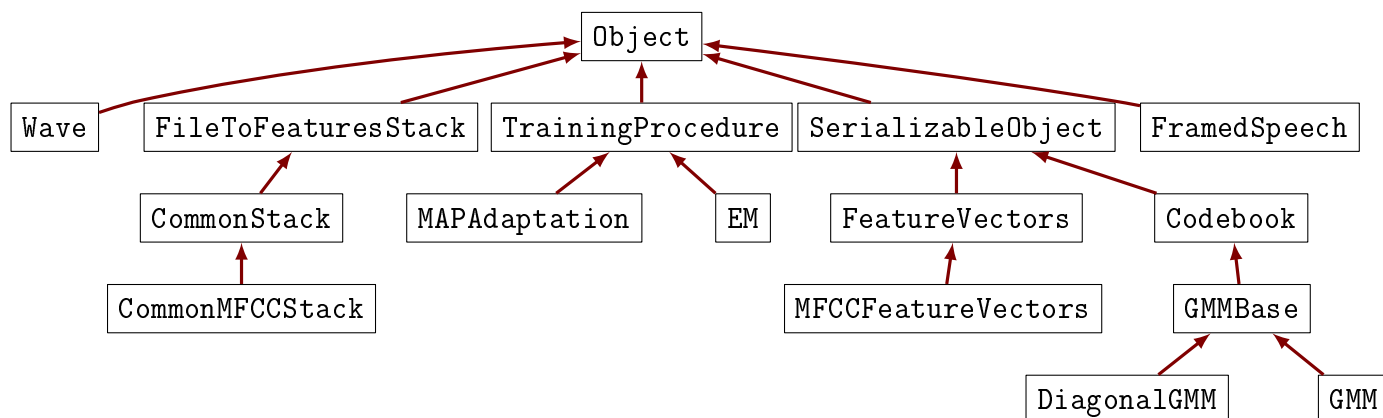


Рисунок 2.10 — Иерархия наследования классов библиотеки голосовой аутентификации

Рассмотрим подробнее введенные классы и их атрибуты:

**Object** : Базовый класс для всех объектов библиотеки. Группирует остальные классы и позволяет управлять глобальным для всех объектов поведением.

**SerializableObject** : класс для объектов, подлежащих сериализации и сохранению в файл. Обладает следующими методами:

- **serialize()**: возвращает бинарное представление объекта в виде строки;
- **serializable()**: возвращает контекст объекта, подходящий для передачи в **serialize()** и последующего восстановления объекта;
- **dump\_to\_file(filename)**: принимает путь к файлу, в который будет сохранен сериализованный объект;
- **deserialize(loaded\_context)**: преобразует контекст, загруженный из файла, в объект данного класса и возвращает этот объект (классовый метод);



- `load(filename)`: принимает путь к файлу, в котором сохранен контекст объекта, возвращает объект (классовый метод);

**Wave** : класс, представляющий звуковую волну как массив амплитуд и значение частоты дискретизации. Используется для преобразования звукового файла одного из поддерживаемых форматов (например, WAV) во внутреннее представление для последующей обработки. Методы и атрибуты класса:

- `read_file(filename)`: конструктор, считывает данные из звукового файла, возвращает объект класса;
- `write_file(filename, format)`: принимает путь к файлу и код формата. Преобразует внутреннее представление звука в файл заданного формата;
- `waveform`: массив амплитуд (цифровое представление звука);
- `samplerate`: частота дискретизации сигнала.

**FramedSpeech** : класс, представляющий сегментированный сигнал. Класс не имеет методов, кроме конструктора, который принимает объект класса **Wave**, длину окна (сегмента) и степень наложения (кадры могут пересекаться друг с другом, таким образом, начало второго кадра совпадает с концом второго и т. д.). Класс имеет единственный атрибут **frames**, представляющий сегментированный сигнал в виде двумерного массива (каждая строка – отдельный сегмент).

**FeatureVectors** : базовый класс для классов, представляющих массивы векторов характерных признаков (по которым происходит обучение потомков класса **Codebook** и последующая аутентификация). Все потомки данного класса обязаны определить метод `get_features()`, возвращающий характерные признаки, по которым обучаются потомки **Codebook**.

**MFCCFeatureVectors** : класс, представляющий массив мел-частотных коэффициентов кепстра, описанных в разделе 1.2.3. Предназначен для выделения признаков из сегментированного сигнала, на вход конструктор получает объект класса **FramedSpeech** и выделяет из каждого сегмента вектор характерных признаков.

**TrainingProcedure** : базовый класс, абстрагирующий процедуру обучения. Описывает метод, который должен быть определен для реализации интерфейса:

- `train(codebook, feature_vectors)`: обучение модели (переданной как параметр `codebook`) по массиву векторов характерных признаков (переданному как `feature_vectors`). Метод возвращает количество итераций (или другую количественную меру длительности обучения).

**EM** : потомок `TrainingProcedure`, реализующий алгоритм Expectation-Maximization для получения параметров модели по обучающей последовательности (см. раздел 1.2.4, [4]). Методы:

- `expectation()`: соответствует шагу E алгоритма EM;
- `maximization()`: соответствует шагу M алгоритма EM;

**MAPAdaptation** : потомок `TrainingProcedure`, реализующий адаптационный алгоритм (Байесово обучение, *max a posteriori adaptation*). Адаптационный алгоритм получает получить персональную GMM-модель с помощью универсальной фоновой модели, заранее обученной на большом количестве речевых данных.

**FileToFeaturesStack** : класс, представляющий полный цикл необходимых преобразований речи. Позволяет объединить в конвейер объекты классов `Wave`, `FramedSpeech`, `FeatureVectors` и скрыть логику получения характерных признаков речи из звукового файла. Методы и атрибуты:

- `process(filename)`: запустить выполнение процедур, добавленных в конвейер. Процедуры выполняются по очереди, при этом результат выполненной процедуры передается в качестве аргумента в следующую по списку процедуру. Возвращается результат последней процедуры в списке;
- `conveyor`: списковая структура, хранящая ссылки на процедуры, которые требуется выполнить по очереди (с помощью метода `process()`);

**CommonStack** : класс, представляющий конвейер преобразований речи, общий для любых характерных признаков (считывание файла и нормализация массива амплитуд). Атрибуты, добавляемые в конвейер:

- `reader`: процедура, преобразующая звуковой файл во внутреннее представление;
- `framer`: процедура, сегментирующая внутреннее представление сигнала;
- `extractor`: процедура, выделяющая характерные признаки из сигнала.

**CommonMFCCStack** : класс, описывающий цепочку преобразований, применяемых к речевому сигналу для получения характерных признаков в виде **MFCCFeatureVectors**, включая процедуры нормализации.

**Codebook** : базовый класс для сущностей, которые могут быть обучены с помощью массива векторов характерных признаков.

**GMMBase** : базовый класс для моделей на основе гауссовых смесей (GMM). Конструктор принимает два параметра:  $K$  — количество компонент смеси,  $D$  — размерность компонент (равна размерности вектора характерных признаков). Определяет интерфейс из следующих необходимых методов:

- **set\_params(w, mu, cov)**: установить параметры модели (параметры, соответственно: веса компонент, двумерный массив векторов математических ожиданий для каждой компоненты и двумерный массив, представляющий матрицы ковариаций компонент);
- **get\_params(w, mu, cov)**: получить параметры модели, возвращает **w**, **mu**, **cov**, описанные выше;
- **likelihood(feature\_vectors)**: принимает объект класса **FeatureVectors**, возвращает значение меры правдоподобия (насколько распределение величин в **FeatureVectors** близко к моделируемому);
- **loglikelihood(feature\_vectors)**: версия **likelihood**, возвращающая логарифм правдоподобия (для облегчения расчетов и борьбы с погрешностью округления);

**GMM** : реализация интерфейса **GMMBase** для модели с полной матрицей ковариаций у каждой из компонент;

**DiagonalGMM** : реализация интерфейса **GMMBase** для модели с диагональной матрицей ковариаций.

Рассмотрим процесс аутентификации с точки зрения потоков данных и функций. На рисунке 2.11 показана диаграмма функциональных блоков для процесса верификации.

- а) Прием речевых данных от пользователя. Данный этап не относится напрямую к ядру системы аутентификации и приведен для полноты представления о процессе. В результате данного этапа, подробно описанного в раз-

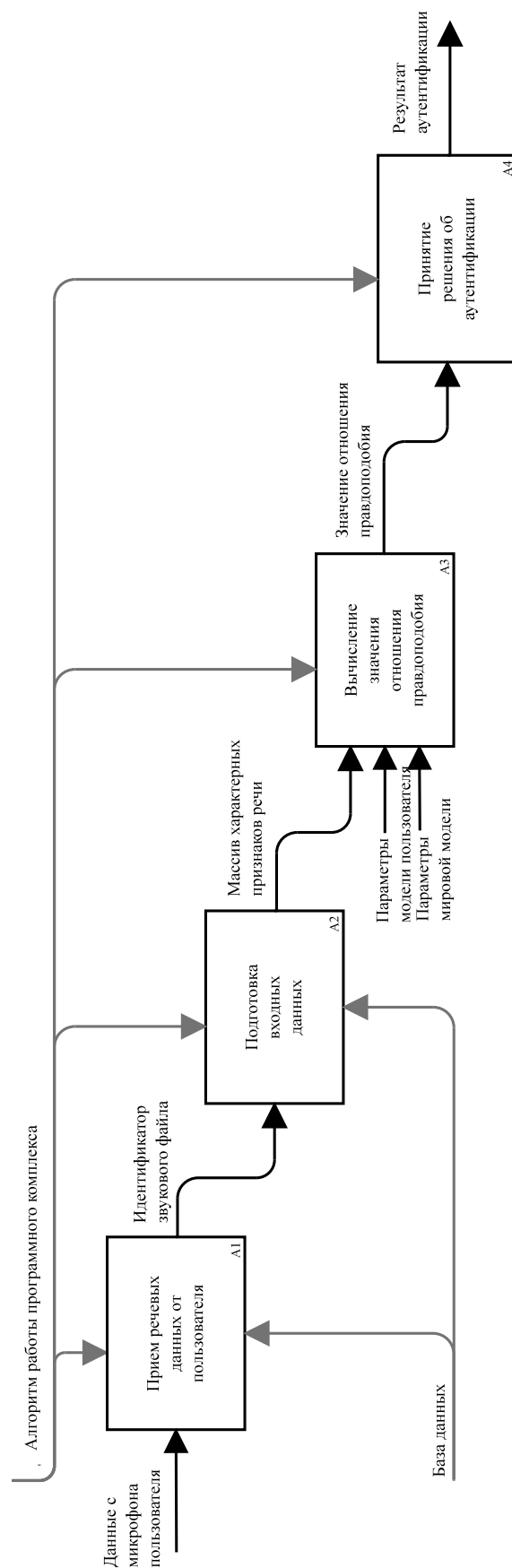


Рисунок 2.11 — Диаграмма функциональных блоков А0: аутентификация по голосу

деле 2.4, на вход ядру системы подается идентификатор звукового файла, который необходимо проверить на аутентичность;

- б) Подготовка входных данных. На данном этапе выполняются все необходимые действия для преобразования звукового файла в массив характерных признаков;
- в) Вычисление значения отношения правдоподобия. По данному массиву характерных признаков рассчитывается правдоподобие основной и альтернативной гипотезы (через функцию правдоподобия для модели пользователя и мировой модели (универсальной фоновой модели));
- г) Принятие решения об аутентификации. Значение отношения сравнивается с порогом вхождения, на основе сравнения принимается решение об аутентификации пользователя.

Блок подготовки входных данных детализирован на рисунке 2.12. Рассмотрим процесс преобразования речи в массив характерных признаков более подробно:

- а) Считывание звукового файла из хранилища. На этом этапе идентификатор звукового файла (его путь в файловой системе) преобразуется в массив амплитуд;
- б) Сегментация речевой последовательности. Массив амплитуд сегментируется и преобразуется в двумерный массив звуковых кадров;
- в) Удаление кадров без речи. По завершении получаем двумерный массив кадров, в котором отсутствуют кадры, содержащие тишину (точнее, внешний шум);
- г) Вычисление мел-частотных кепстральных коэффициентов. На этом этапе каждый сегмент сигнала преобразуется в массив кепстральных коэффициентов в соответствии с описанием, данным в разделе 1.2.3.
- д) Расчет дельта-коэффициентов. В соответствии с выражением (1.2), для каждого из коэффициентов MFCC рассчитывается дельта-коэффициент, таким образом, размерность вектора характерных признаков увеличивается вдвое. Это завершающий этап обработки звукового файла, в результате которого получается массив векторов характерных признаков, используемый в процессе обучения и аутентификации.

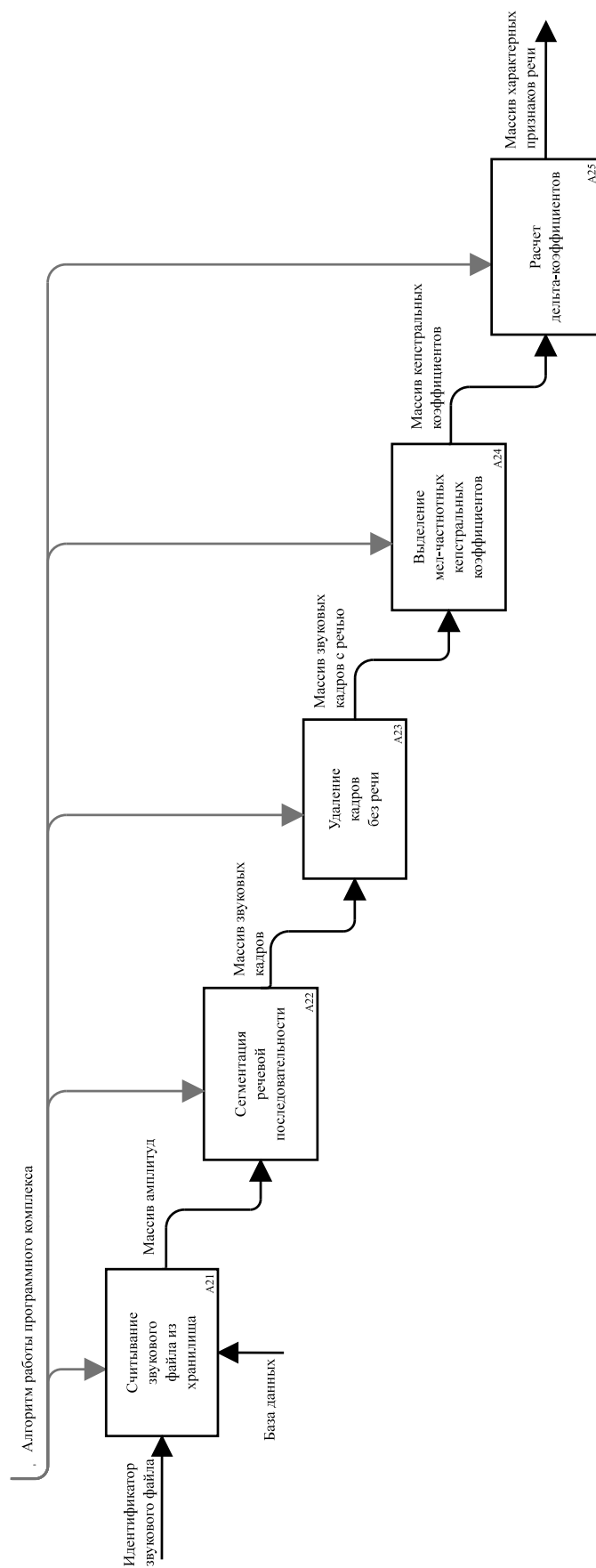


Рисунок 2.12 — Диаграмма функциональных блоков A1: подготовка входных данных

## 2.7 Алгоритм удаления тишины

Как для процесса обучения, так и для процесса аутентификации, для достижения адекватных показателей ошибок необходима нормализация входных речевых данных. Одним из наиболее важных этапов при применении гауссовых смесей<sup>1</sup> является выделение из речевых данных только тех, которые вероятнее содержат речь, а не внешний шум.

Для отделения во входном сигнале речи от условной тишины (постоянного внешнего шума, создаваемого окружением) был разработан специализированный алгоритм, блок-схема которого показана на рисунке 2.13. Алгоритм основан на двух предположениях:

- а) В начала запись всегда есть отрезок без речи. Это предположение равнозначно предположению о том, что человек реагирует на начало записи не мгновенно, а некоторым опозданием. Величина этого отрезка является одним из входных параметров алгоритма;
- б) Амплитуда шума имеет распределение, близкое к нормальному. Это предположение исходит из того, что шум окружения – это случайная величина, зависящая от большого количества факторов, каждый из которых вносит небольшой вклад. Однако, даже если данное предположение неверно, коэффициент доверия остается в пределах, позволяющих практическое использование алгоритма.

Суть алгоритма проста: предполагая, что первые  $\tau_T$  миллисекунд записи – это внешний шум (условная тишина), рассматриваем амплитуду сигнала как случайную величину и рассчитываем параметры:  $\mu$  – выборочное среднее и  $\sigma$  – выборочное средне-квадратичное отклонение (несмещенная оценка). Тогда, заменяя точные значения выборочными, можно использовать правило, известное в статистике как «правило трех  $\sigma$ » [14]:

$$\mathbf{P}\{|X - \mu| > 3\sigma\} < \frac{\sigma^2}{(3\sigma)^2} = \frac{1}{9} = 0.1(1), \quad (2.1)$$

являющееся формой второго неравенства Чебышева [21].

---

<sup>1</sup>Причиной важности является текст-независимость метода. Паузы между словами и предложениями можно рассматривать как характерные признаки источника речи, однако, используемый метод предназначен для моделирования характеристик вокального тракта человека, а не столь высокоуровневых признаков.

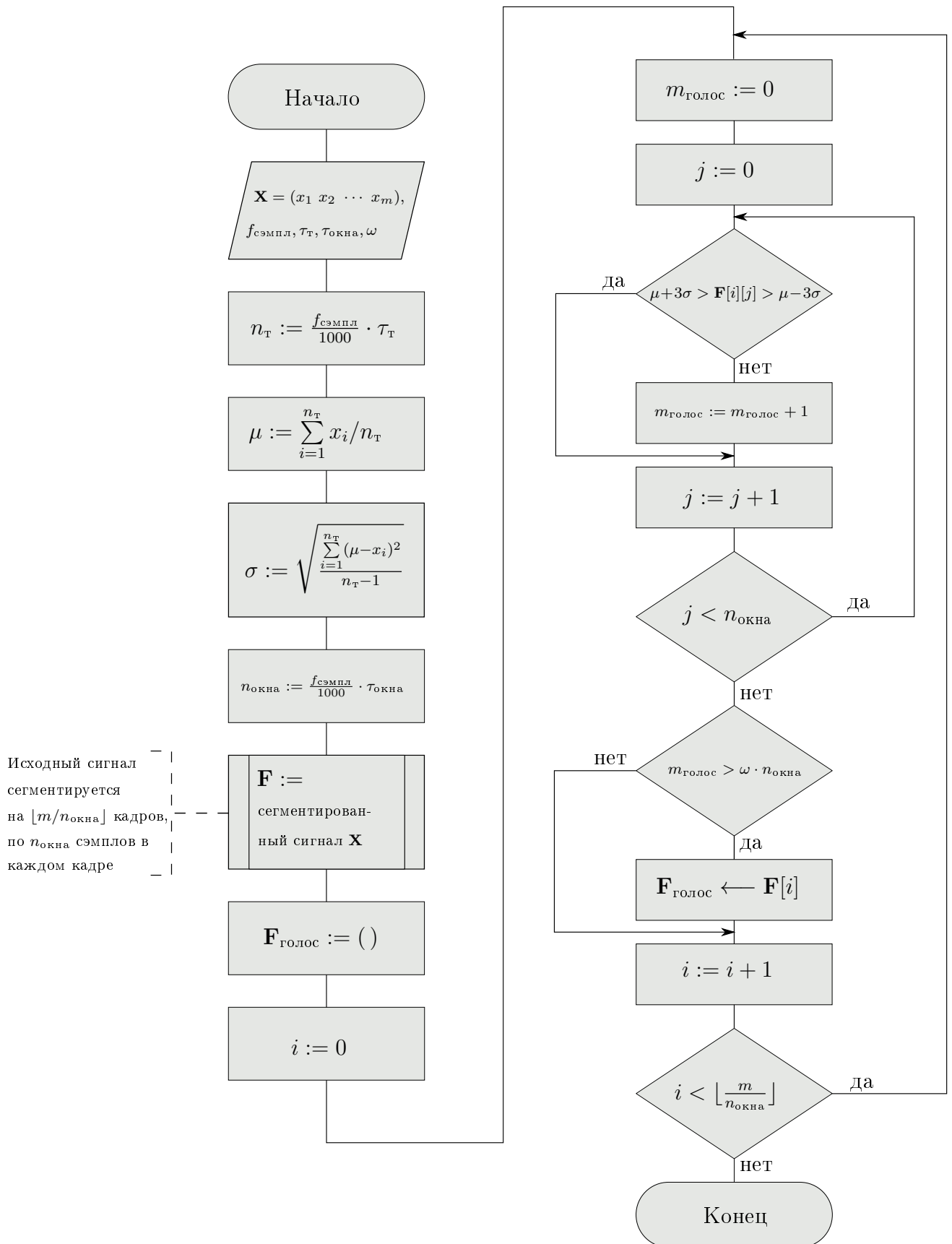


Рисунок 2.13 — Блок-схема алгоритма удаления «тишины» из входного сигнала



В случае нормального распределения, неравенство превращается в равенство:

$$\mathbf{P}\{|X - \mu| < 3\sigma\} = F(\mu + 3\sigma) - F(\mu - 3\sigma) = \Phi(3) - \Phi(-3) = 0.9973, \quad (2.2)$$

которое можно неформально интерпретировать так: «99.73% значений случайной величины  $X$  находятся в интервале  $(\mu - 3\sigma, \mu + 3\sigma)$ ». Входные данные алгоритма описаны в таблице 2.1.

Таблица 2.1 — Входные данные алгоритма удаления тишины

| Имя переменной                      | Описание  |
|-------------------------------------|---|
| $\mathbf{X} = (x_1 x_2 \cdots x_m)$ | Входной сигнал (массив амплитуд) длиной $m$   |
| $f_{\text{сэмпл}}$                  | Частота сэмплирования (дискретизации), Гц   |
| $\tau_{\text{T}}$                   | Длина заведомо не содержащего речи отрезка записи (эмпирическая величина), мс   |
| $\tau_{\text{окна}}$                | Ширина окна для сегментирования, мс   |
| $\omega$                            | Параметр, регулирующий чувствительность алгоритма (0 – алгоритм вернет исходный сигнал, 1 – максимальная чувствительность), действительное число в интервале $(0, 1]$ |

Рассмотрим основные этапы алгоритма:

- а) определяем количество сэмплов  $n_{\text{T}}$ , в которых предположительно нет речи;
- б) определяем выборочное среднее  $\mu$  (как среднее арифметическое по первым  $n_{\text{T}}$  сэмплам сигнала);
- в) определяем выборочную дисперсию  $\sigma$  (несмещенную оценку) по первым  $n_{\text{T}}$  сэмплам сигнала;
- г) определяем длину окна  $n_{\text{окна}}$  в сэмплах;
- д) вызываем процедуру сегментации, получаем двумерный массив  $\mathbf{F}$ , представляющий собой исходный сигнал, поделенный на сегменты, в каждом из которых  $n_{\text{окна}}$  сэмплов;
- е) создаем изначально пустую списковую структуру  $\mathbf{F}_{\text{голос}}$ , в которую будут добавляться сегменты из  $\mathbf{F}$ , содержащие голосовые данные;
- ж) для каждого сегмента из  $\mathbf{F}$  подсчитываем количество сэмплов  $m_{\text{голос}}$ , для которых выполняется двойное неравенство  $\mu + 3\sigma > \mathbf{F}[i][j] > \mu - 3\sigma$ ;

- з) если доля сэмплов данном сегменте, для которых выполняется предыдущее неравенство, превышает заданную чувствительность  $\omega$ , помечаем сегмент как содержащий голос, добавляем его в  $\mathbf{F}_{\text{голос}}$ ;
- и) повторив действия ж)–з) для каждого сегмента из  $\mathbf{F}$ , возвращаем  $\mathbf{F}_{\text{голос}}$ .

В результате выполнения алгоритма, получается сегментированный сигнал, содержащий только те сегменты, которые удовлетворяют наложенным условиям. После этого сигнал можно снова растянуть и работать как с исходным. Влияние применения разработанного алгоритма на качество аутентификации, а также зависимость качества от параметра  $\omega$  рассмотрено в разделе 4.

## 2.8 Выводы

В результате выполнения конструкторской части проекта можно сделать следующие выводы:

- а) спроектирована общая архитектура программного комплекса голосовой аутентификации пользователя;
- б) спроектирована подсистема регистрации пользователя;
- в) спроектирована подсистема аутентификации пользователя;
- г) спроектирована схема базы данных и набор сущностей программного комплекса;
- д) спроектирована библиотека функций, предназначенных для обработки речи, создания моделей на основе гауссовых смесей и проведения аутентификации. Библиотека разработана с учетом возможности дополнения и расширения функционала;
- е) построен алгоритм удаления фрагментов записи, не содержащих речь (удаления тишины).

## 3 Технологический раздел

### 3.1 Выбор среды разработки

В качестве основной технологии для разработки серверной части системы выбран язык высокого уровня **Python** и набор инструментов для построения веб-приложений **Django**.

На выбор языка оказали влияние следующие факторы:

- а) ясность и читаемость кода как следствие синтаксических особенностей языка;
- б) возможность создания модульных программ, поддержка иерархии пакетов;
- в) поддержка основных возможностей функционального программирования, а также полная поддержка парадигмы ООП;
- г) высокоуровневые типы данных;
- д) динамическая типизация;
- е) большое количество стандартных библиотек и функций;
- ж) язык адаптирован для прототипирования;
- з) наличие библиотек для работы с численными массивами и матричных вычислений (NumPy, SciPy), а также дополнительных утилит для обработки сигналов, позволяет использовать Python в качестве альтернативы таким системам, как MatLab;
- и) свободная лицензия позволяет использовать язык для создания как свободных, так и проприетарных приложений.

На выбор **Django** для построения серверной части комплекса повлияли следующие факторы:

- а) использование данного ПО для реализации целевой системы (системы управления студенческими проектами кафедры ИУ-7 МГТУ им. Н. Э. Баумана);
- б) использование **Python** в качестве языка программирования. **Python** предоставляет мощные средства метапрограммирования, обширную библиотеку классов, хорошую документацию;

- в) полная документация. Вместе с **Django** поставляется качественная документация с подробным объяснением основных концепций, множество примеров;
- г) встроенный ORM (*Object-relational mapper*). Обеспечивает проецирование реляционных данных в объекты, благодаря чему в большинстве случаев совершенно не требуется использование SQL-синтаксиса в выражениях, что автоматически снижает риск появления **SQL-injection** уязвимости;
- д) поддержка MTV (**Model-Template-View**). Данный паттерн проектирования очень близок к классическому MVC, позволяет отделять бизнес-логику от дизайна;
- е) высокая скорость работы. **Django** может выдерживать высокую нагрузку, так как имеет встроенные средства кэширования и распределения нагрузки;
- ж) предоставление прототипов для развертывания административной части системы.

Для реализации очередей запросов выбран протокол **AMQP** (Advanced Message Queuing Protocol) и его серверная реализация на языке **Erlang** — **RabbitMQ**.

Для реализации функционала на стороне клиента использован HTML/JavaScript. Для работы с микрофоном, записи и отправки голосовых данных на сервер используется Java-апплет. Решено использовать Java по следующим причинам:

- а) на сегодняшний день единственной альтернативой встраивания Java для доступа к микрофону пользователя из браузера является технология Flash или создание зависящих от браузера дополнений;
- б) при использовании Flash появляется необходимость использования проприетарного приложения Flash Media Server для получения данных от пользователя;
- в) среда исполнения Java доступна на большинстве существующих программно-аппаратных платформ, в том числе мобильной платформе Windows Mobile 6, апплеты широко распространены, поэтому снижается требовательность системы к конфигурации пользовательской ЭВМ.

## 3.2 Описание программы

### 3.2.1 Общие сведения

Для функционирования программы необходимо следующее программное обеспечение:

- Python 2.6.2;
- SciPy 0.7.2 с набором инструментов **scikits** (<http://scikits.appspot.com>, 2010);
- NumPy 1.4.1;
- Django 1.1.1;
- Apache 2.2.14 с модулем `mod_python` или `wsgi` (или другой веб-сервер, поддерживающий интерфейс WSGI);
- RabbitMQ 1.5.4 (<http://www.rabbitmq.com>, 2010);
- GNU Make 3.81;
- Поддерживаемая Django СУБД: PostgreSQL версии 8.2 или выше, MySQL 4.1/5.0, Oracle версии 9i или выше, SQLite версии 3.3.6 или выше; возможно использование других СУБД при наличии соответствующего интерфейса (см. <http://docs.djangoproject.com/en/dev/ref/databases/#using-a-3rd-party-database-backend>, 2010);

Программа написана полностью на языке **Python** (<http://www.python.org>, 2010) с использованием веб-фреймворка **Django** (<http://www.djangoproject.com>, 2010), библиотеки для работы с многомерными массивами **NumPy** и библиотеки научных инструментов **SciPy** (<http://www.scipy.org>, 2010). Все вышеперечисленное ПО (за исключением СУБД Oracle) является свободным и кроссплатформенным.

### 3.2.2 Функциональное назначение

Программный комплекс предназначен для голосовой аутентификации пользователя интернет-сайта.

Основные функциональные возможности:

- создание персональной модели голоса пользователя, позволяющей аутентифицировать его по ключевой фразе;

- вход в защищаемый раздел интернет-ресурса по ключевой фразе, используемой при регистрации;
- возможность добавления, просмотра и редактирования параметров и сущностей приложения через интерфейс администратора;
- ведение системного журнала событий;
- возможность интеграции в существующий интернет-портал, использующий технологию Django.

### 3.2.3 Описание логической структуры

В данном разделе дано описание основных модулей программного комплекса.

#### Структура модулей системы

На рисунке 3.1 представлена иерархия модулей системы по отношению взаимного использования. Модули пакета `verispeak`, который представляет собой реализацию библиотеки инструментов для голосовой аутентификации, детально рассмотрены далее.

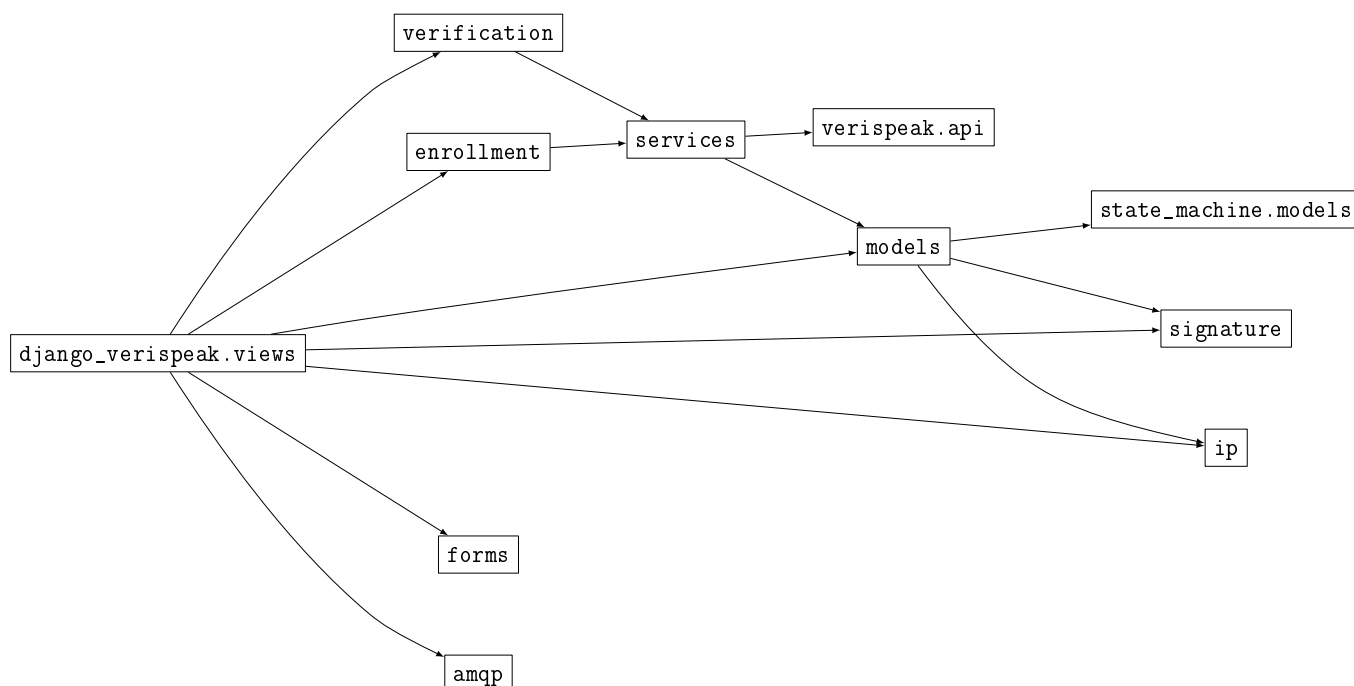


Рисунок 3.1 — Граф импортирования модулей системы (контроллер)

Модули контроллера соответствуют соглашениям и требованиям используемого инструментария **Django**:

- **views** — модуль, содержащий определения всех ресурсов системы аутентификации, предоставляемых сервером внешним клиентам;
- **models** — модуль, определяющий сущности системы в виде классов, которые отображаются в базу данных;
- **forms** — модуль, содержащий классы, которые представляют HTML-формы;
- **views.enrollment** — подмодуль, содержащий определения ресурсов, относящихся к стадии регистрации (обучения модели);
- **views.verification** — подмодуль, содержащий определения ресурсов, относящихся к стадии аутентификации;
- **services** — в данном модуле определены обработчики заявок на обучение модели и аутентификацию. Это единственный модуль контроллера, который связан с библиотекой функций для голосовой аутентификации (через `verispeak.api`);
- **amqp** — клиент для протокола обмена сообщениями AMQP, позволяющий ставить заявки на обучение модели и аутентификацию, код обработчиков которых находится в модуле **services**;
- **ip** — вспомогательный модуль, используемый для преобразования строкового представления IP-адреса клиента в упакованный вид (целое число) и обратно;
- **signature** — вспомогательный модуль, определяющий функции для подписывания данных секретным ключом (используется при генерации уникального идентификатора сессии в сессиях регистрации и аутентификации);
- **state\_machine.models** — в данном модуле определен базовый класс для реализации сущностей, обладающих состояниями (процесс аутентификации и процесс обучения модели).

Опишем модули библиотеки инструментов для голосовой аутентификации, граф импортирования которых представлен на рисунке 3.2:

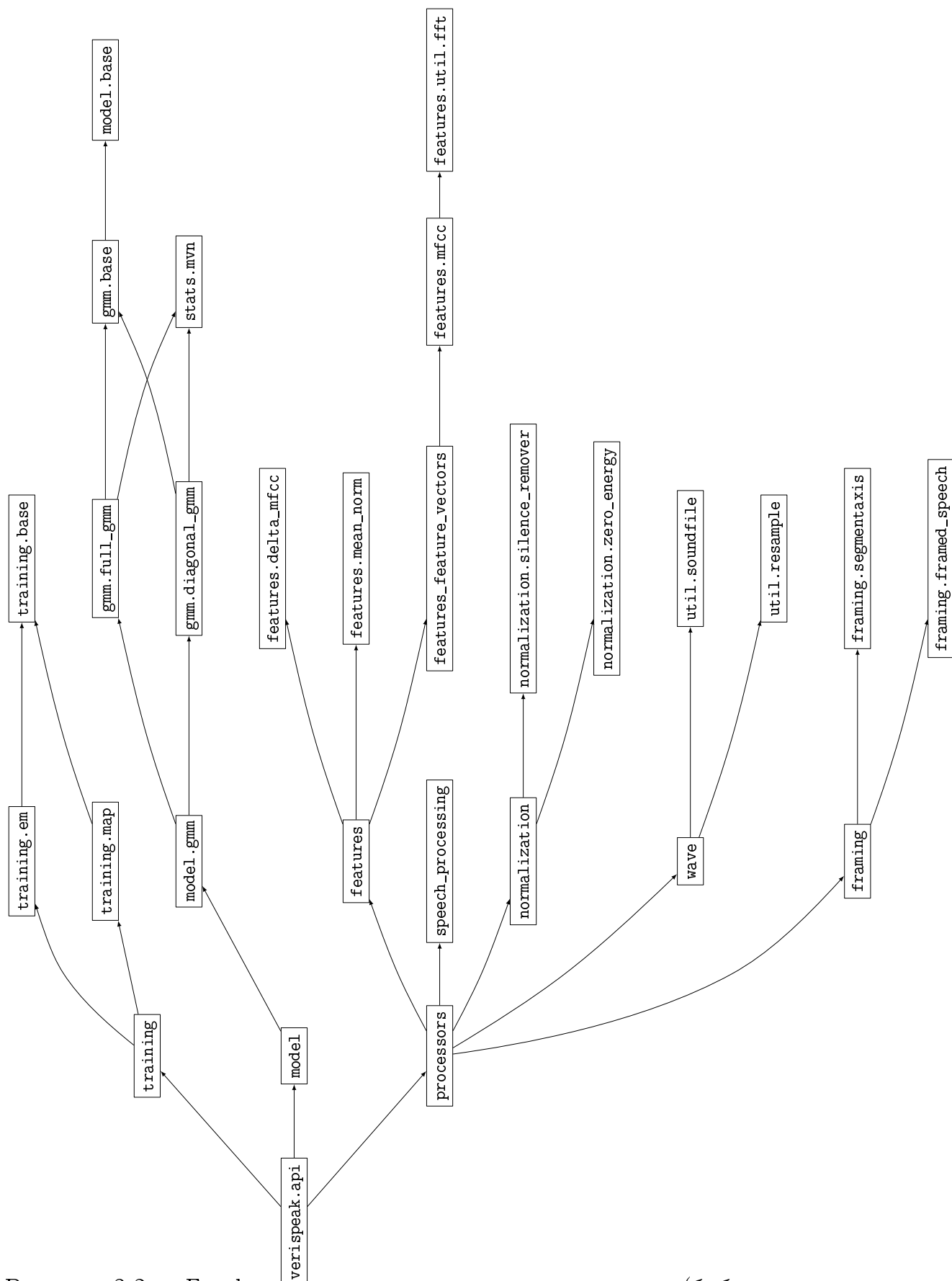


Рисунок 3.2 — Граф импортирования модулей системы (библиотеки голосовой аутентификации)



- `model` — модуль содержит определения классов, предназначенных для моделирования источника речи;
- `model.base` — модуль, в котором содержится определение базового класса `Codebook`;
- `model.gmm` — модуль содержит реализации `Codebook`, основанные на гауссовых смесях;
- `model.gmm.base` — определение базового класса `GmmBase`;
- `model.gmm.full_gmm` — определение класса `FullGMM`;
- `model.gmm.diagonal_gmm` — определение класса `DiagonalGMM`;
- `stats.mvn` — модуль содержит различные функции для работы с многомерным нормальным распределением: нахождение значения функции плотности распределения в некоторой точке многомерного пространства;
- `training` — модуль содержит определения классов, предназначенных для обучения моделей (модуль `model`) по массивам характерных признаков речи (модуль `features`);
- `training.base` — определение базового класса `TrainingProcedure`;
- `training.em` — определение класса `EM`;
- `training.map` — определение класса `MapAdaptation`;
- `processors` — определение классов `CommonStack`, `CommonMFCCStack`;
- `speech_processing` — определение класса `FileToFeaturesStack`;
- `features` — модуль содержит различные классы и функции, предназначенные для выделения характерных признаков речи из звукового сигнала;
- `features.feature_vectors` — определение класса `FeatureVectors`;
- `features.mean_norm` — определение функции, реализующей нормализацию массива векторов характерных признаков по средним;
- `features.delta_mfcc` — определение функции, реализующей расчет дельта-коэффициентов для векторов характерных признаков;
- `features.mfcc` — определение класса `MFCCFeatureVectors`;
- `wave` — определение класса `Wave`;
- `util.soundfile` — модуль содержит функции для считывания с диска и записи на диск звуковых файлов в формате WAV;

- `util.resample` — модуль содержит функции для изменения частоты дискретизации оцифрованного сигнала;
- `normalization` — модуль содержит различные функции для нормализации звукового сигнала;
- `normalization.silence_remover` — модуль содержит функцию, реализующую алгоритм удаления тишины (раздел 2.7);
- `normalization.zero_energy` — модуль содержит функцию, предназначенную для удаления из сегментированного сигнала кадров со значением энергии ниже порогового;
- `framing.framed_speech` — определение класса `FramedSpeech`;
- `framing.segment_axis` — модуль содержит функцию, предназначенную для сегментации входного сигнала.

### **Физическое расположение компонент системы**

На рисунке 3.3 представлена диаграмма внедрения, соответствующая общей архитектуре системы, описанной в разделе 2.1.

#### **3.2.4 Используемые технические средства**

В состав технических средств должна входить ЭВМ, включающая в себя:

- процессор архитектуры, поддерживаемой UNIX-подобными ОС: `amd64`, `i386`, `ia64`, `MIPS`, `ARM`, `ppc`, `sparc64`, `sun4v`, `xbox`;
- оперативную память объемом не менее 64МБ;
- не менее 30МБ свободного места на диске для установки комплекса, а также дополнительное место для базы данных (не менее 50МБ, а также не менее 1МБ на каждого регистрируемого пользователя).

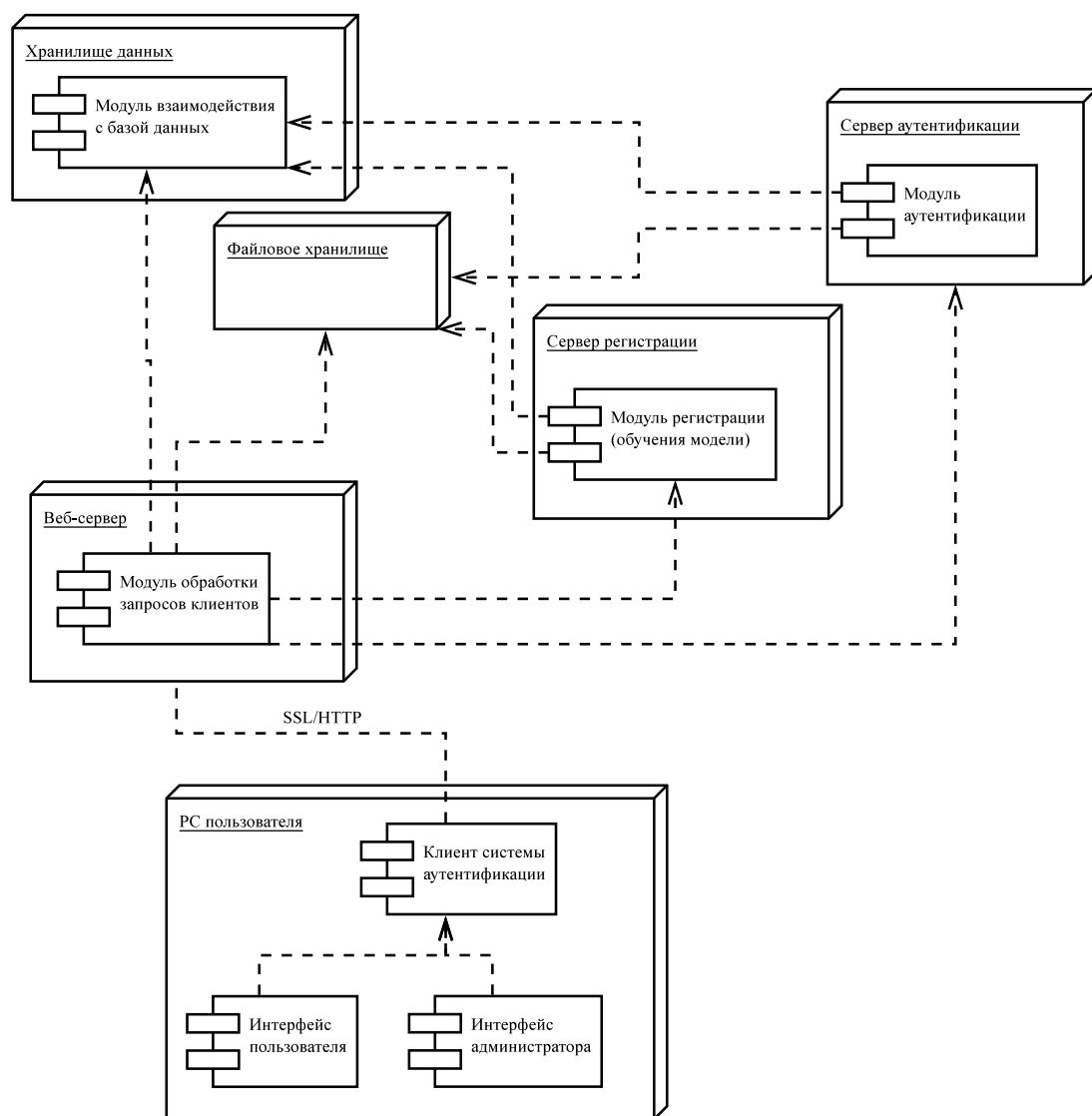


Рисунок 3.3 — Диаграмма внедрения

### 3.2.5 Вызов и загрузка

Для установки комплекса на сервер необходимо выполнить следующие команды из каталога проекта:

```
make build
```

```
make install
```

Команда `make build` выполнит все необходимые действия для сборки готового пакета, `make install` – все необходимые действия по созданию работающего веб-приложения, конфигурации базы данных и веб-сервера. После выполнения данных команд система сразу начинает работу. Во время выполнения скрипта при необходимости могут быть запрошены данные, например, пути к каталогам, в которых будут размещены архивы проектов, а также имя виртуального хоста, которое используется при конфигурации Apache. После успешного завершения установки, интерфейс администратора доступен по относительному URL-адресу `/admin/`, страница приветствия системы аутентификации – по адресу `/voice/`.

## 3.3 Руководство пользователя

### 3.3.1 Назначение программы

Программа предназначена для осуществления аутентификации пользователя интернет-сайта по голосу.

### 3.3.2 Условия выполнения программы

Минимальный состав технических средств описан в разделе 3.2.4.

Минимальный состав программных средств описан в разделе 3.2.1.

### 3.3.3 Выполнение программы

На рисунке 3.4 представлена главная страница системы голосовой аутентификации. Справа расположен логотип системы, под которым отражается статус пользователя (авторизован или нет). В центральной части страницы находится ссылка для перехода на страницу аутентификации и форма ввода имени учетной

записи. В нижней части представлена «Навигация», которая состоит из ссылки на регистрацию в системе и ссылки для перехода на главную страницу защищаемого интернет-ресурса. Если пользователь не авторизован на интернет-ресурсе, то в списке «Навигация» присутствует ссылка на страницу с формой ввода имени учетной записи и пароля. Если пользователь авторизован, то вместо ссылки на страницу с формой ввода имени учетной записи и пароля присутствует ссылка «Выход».


**Вас приветствует система голосовой аутентификации пользователя!**

Добро пожаловать на страницу системы голосовой аутентификации!

- [Вход в систему по голосу](#) (имя учетной записи: )

**Навигация**

- [Главная страница](#)
- [Регистрация в системе](#)



Вы не авторизованы

Рисунок 3.4 — Главная страница системы голосовой аутентификации

При выборе пункта «Регистрация в системе» пользователь автоматически переходит на страницу, которая содержит список требований, удовлетворение которых необходимо для регистрации в системе. Если пользователь авторизован на интернет-ресурсе, на странице присутствует ссылка для перехода к созданию персональной голосовой модели (рисунок 3.5). Если пользователь не авторизован, на странице отображается сообщение о необходимости авторизации (рисунок 3.6).

### 3.3.3.1 Создание индивидуальной голосовой модели

В случае выбора ссылки «Перейти к созданию индивидуальной голосовой модели», производится переход к странице с формой, которая представлена на рисунке 3.7. В центральной части формы находится краткая инструкция по записи образца ключевой фразы. В нижней части находится кнопка «Возврат», при выборе которой происходит автоматический переход на главную страницу системы голосовой аутентификации, и кнопка «Запись».

## Регистрация в системе голосовой аутентификации

Чтобы иметь возможность производить аутентификацию по голосу, Вам необходимо пройти регистрацию в системе. Для этого необходимо иметь:

- Интернет-обозреватель, поддерживающий Java-апплеты (на официальном сайте Sun Microsystems можно [проверить](#), поддерживаются ли апплеты в Вашем обозревателе);
- Существующую учетную запись на сайте (Вы зашли как n.zakharov);
- Работаящий микрофон, подключенный к Вашему ПК.



Вы вошли как n.zakharov

[Перейти к созданию персональной голосовой модели.](#)

### Навигация

- [Главная страница](#)
- [Регистрация в системе](#)
- [Выход](#)

Рисунок 3.5 — Информация о регистрации (пользователь авторизован)

## Регистрация в системе голосовой аутентификации

Чтобы иметь возможность производить аутентификацию по голосу, Вам необходимо пройти регистрацию в системе. Для этого необходимо иметь:

- Интернет-обозреватель, поддерживающий Java-апплеты (на официальном сайте Sun Microsystems можно [проверить](#), поддерживаются ли апплеты в Вашем обозревателе);
- Существующую учетную запись на сайте ([Регистрация](#));
- Работаящий микрофон, подключенный к Вашему ПК.



Вы не авторизованы

Вы не вошли на сайт. Для продолжения, заведите учетную запись на сайте и авторизуйтесь.

### Навигация

- [Главная страница](#)
- [Регистрация в системе](#)
- [Вход](#)

Рисунок 3.6 — Информация о регистрации (пользователь не авторизован)

При выборе кнопки «Запись» начинается запись, и форма автоматически переходит в состояние, изображенное на рисунке 3.8. В центральной части формы отражается состояние времени записи. В нижней части, заменяя кнопку «Запись»,

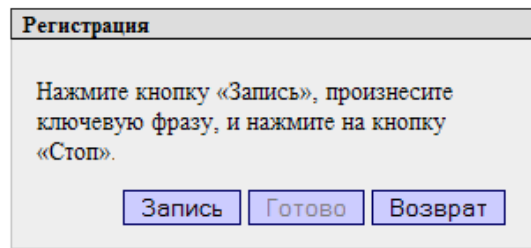


Рисунок 3.7 — Регистрация: инициализация сессии записи

появляется кнопка «Стоп», при нажатии на которую происходит остановка записи.

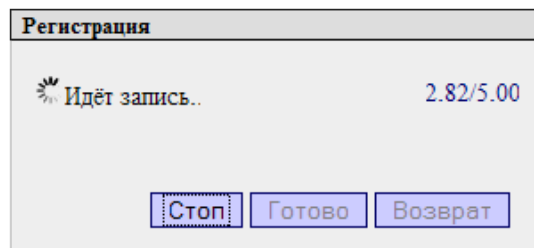


Рисунок 3.8 — Регистрация: запись голосового образца

Если записанный образец слишком короткий, то при выборе кнопки «Стоп», форма переходит в состояние, представленное на рисунке 3.9. В центре формы находится сообщение о длительности образца и рекомендации пользователю. Из данной формы можно перейти в состояние записи голосового образца или вернуться на главную страницу системы голосовой аутентификации.

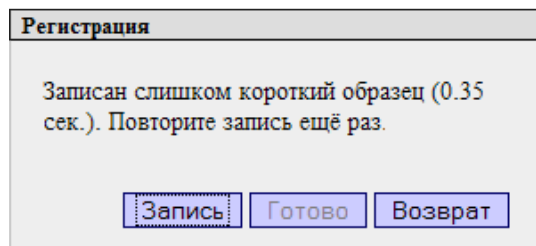


Рисунок 3.9 — Регистрация: недостаточно данных для отправления

Если образец имеет достаточную длину, форма (рисунок 3.8) переходит в состояние, представленное на рисунке 3.10. В этом состоянии происходит отправление записи, что отражается в центральной части формы, также там отражено время последней записи.

При возникновении ошибки во время отправления записи, форма переходит в состояние, представленное на рисунке 3.11. В центральной части формы расположено сообщение об ошибке и рекомендации пользователю.

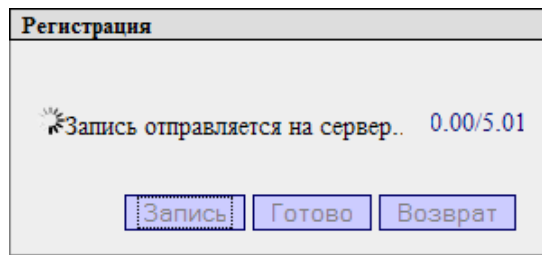


Рисунок 3.10 — Регистрация: отправление записи

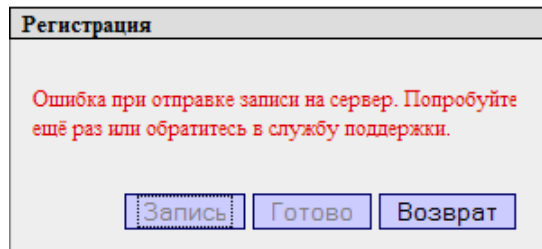


Рисунок 3.11 — Регистрация: ошибка при отправлении записи

При удачном завершении отправления записи на сервер, форма (рисунок 3.10) переходит в состояние, изображенное на рисунке 3.12. На форме отражается количество, общая длительность записей, и сообщение о состоянии записи. Если пользователь хочет отменить процесс регистрации, необходимо выбрать кнопку «Возврат». Если пользователь хочет продолжить создание голосовой модели, необходимо выбрать кнопку «Запись». Когда пользователь запишет достаточное количество образцов ключевой фразы, становится активной кнопка «Готово».

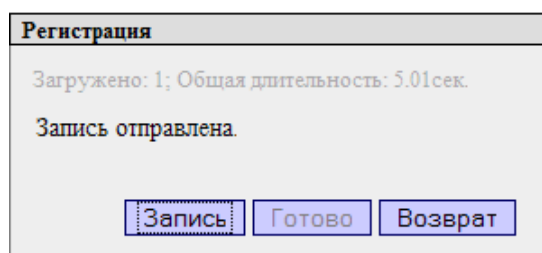


Рисунок 3.12 — Регистрация: запись отправлена

При выборе кнопки «Готово» начинается процесс обучения модели, и форма переходит в состояние, представленное на рисунке 3.13. В центральной части формы расположено сообщение о состоянии процесса обучения.

Если произошла ошибка при обучении модели, форма переходит в состояние, отраженное на рисунке 3.14. В центре формы отображается сообщение об ошибке и рекомендации пользователю. При возникновении данной ошибки, поль-



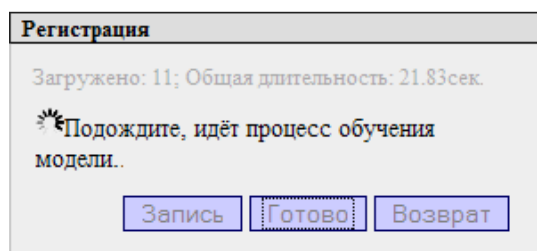


Рисунок 3.13 — Регистрация: обучение модели

зователь может вернуться на главную страницу системы голосовой аутентификации.

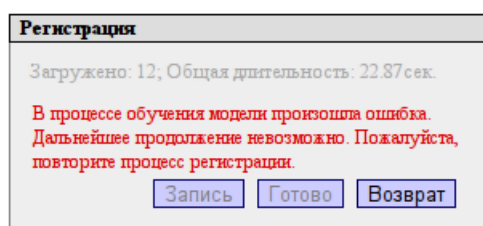


Рисунок 3.14 — Регистрация: ошибка во время обучения модели

Если данных для обучения недостаточно, форма (рисунок 3.13) переходит в состояние, показанное на рисунке 3.15. В центральной части формы отображается сообщение об ошибке и рекомендации пользователю. При возникновении данной ошибки, пользователь может выбрать кнопку «Возврат» или записать дополнительные голосовые данные, выбрав кнопку «Запись».

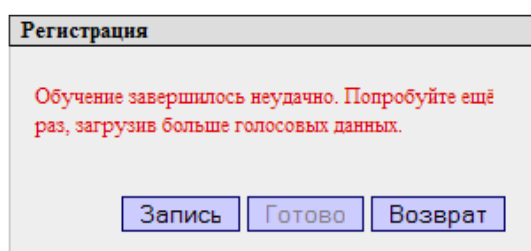


Рисунок 3.15 — Регистрация: недостаточно данных для обучения модели

При удачном завершении обучения модели, регистрация завершается, и форма (рисунок 3.13) переходит в состояние, показанное на рисунке 3.16. Из этого состояния происходит переход на главную страницу системы голосовой аутентификации, что отражается в сообщении.

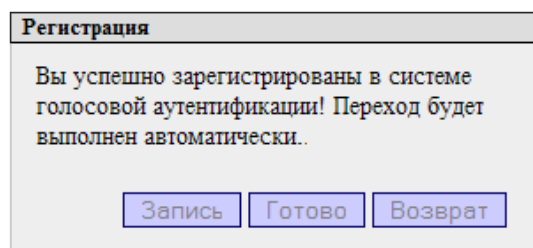


Рисунок 3.16 — Регистрация: успешное завершение обучения модели

### 3.3.3.2 Аутентификация по голосу

Если на главной странице системы голосовой аутентификации не введено имя учётной записи, то при переходе по ссылке «Вход в систему по голосу», возникает окно с предупреждением (рисунок 3.17). Если имя учётной записи введено, производится переход к странице с формой, которая представлена на рисунке 3.18. Центральная часть формы и функция кнопки «Запись» аналогичны описанным в разделе 3.3.3.1 (рисунок 3.7). В нижней части формы находится кнопка «Возврат», при выборе которой, производится переход на главную страницу защищаемого интернет-ресурса.

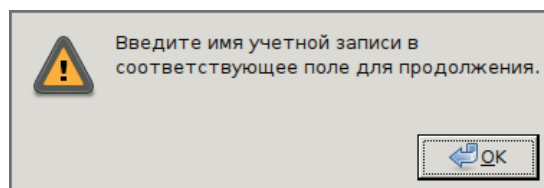


Рисунок 3.17 — Предупреждение о необходимости ввода имени учётной записи

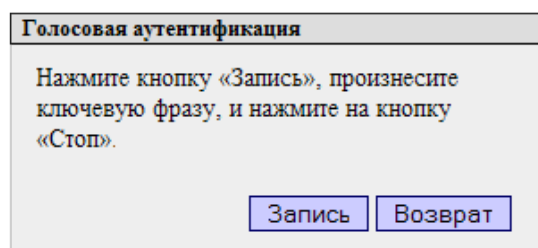


Рисунок 3.18 — Аутентификация: инициализация сессии записи

Процесс записи ключевой фразы для аутентификации аналогичен процессу записи ключевых фраз для создания индивидуальной голосовой модели. Различие заключается в том, что запись производится один раз, после чего начинается процесс аутентификации (рисунок 3.19).

Если во время аутентификации произошла ошибка, форма переходит в состояние, отраженное на рисунке 3.20. В центре формы отображается сообщение об

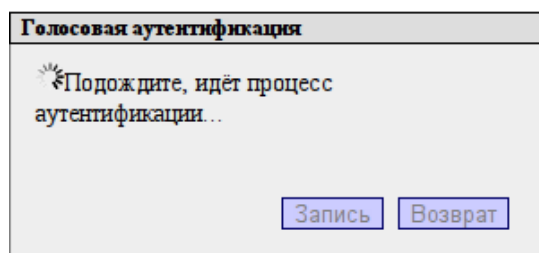


Рисунок 3.19 — Аутентификация: процесс аутентификации

ошибке и рекомендации пользователю. При возникновении данной ошибки, пользователь может повторить попытку аутентификации, выбрав кнопку «Повтор», или перейти на главную страницу интернет-ресурса, выбрав кнопку «Возврат».

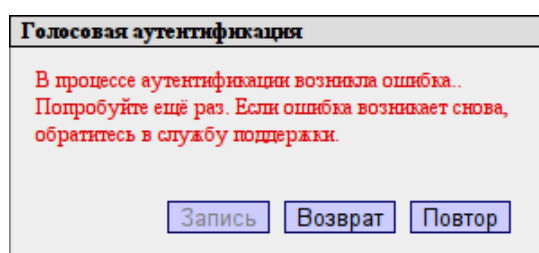


Рисунок 3.20 — Аутентификация: ошибка во время аутентификации

Если процесс аутентификации завершился отказом в доступе, форма (рисунок 3.19) переходит в состояние, показанное на рисунке 3.21. В центре формы отображается сообщение о состоянии аутентификации и рекомендации пользователю. В нижней части формы находятся кнопки «Возврат» и «Повтор».

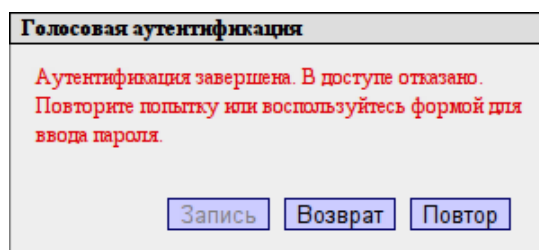


Рисунок 3.21 — Аутентификация: в доступе отказано

Если аутентификация завершается успешно для пользователя (доступ разрешён), происходит автоматический переход на запрошенную страницу, или на страницу, заданную по-умолчанию, администратором интернет-ресурса.

### 3.3.3.3 Интерфейс администратора

На главной странице интерфейса администратора представлен список разделов, отвечающих за управление соответствующими объектами системы (рисунок 3.22). Все перечисленные объекты подробно описаны в разделе 2.5.

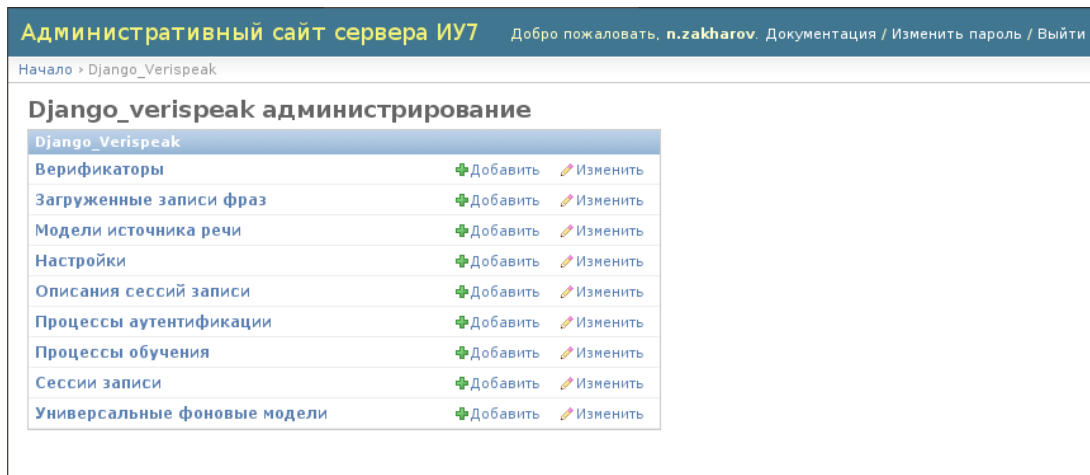


Рисунок 3.22 — Главное окно интерфейса администратора

## 4 Экспериментальный раздел

В данном разделе описаны проведенные эксперименты, призванные определить точность разработанной системы при аутентификации пользователей, а также определить необходимые для работы системы параметры.

### 4.1 Планирование эксперимента

#### 4.1.1 Определение точности системы аутентификации

При анализе биометрических систем, точность системы определяется вероятностями совершения ошибок I-ого и II-ого рода:

- а) аутентификация завершилась неудачно для аутентичного источника речи (ложный отказ);
- б) аутентификация прошла успешно, но источник речи в действительности не аутентичен зарегистрированному (несанкционированный вход).

Требуемый баланс между этими показателями достигается путем варьирования значения порога вхождения. Частота ошибок (*Equal Error Rate*) – значение вероятности при таком пороге вхождения, когда вероятности ошибок I-ого и II-ого рода равны.

Для более подробного отражения качественных характеристик системы используются так называемые DET-кривые (Detection Error Trade-off curves) [3]. На оси  $x$  DET-кривой отражается процент ложного отказа, при этом на оси  $y$  отражается процент ложного положительного решения (при одинаковом пороге). Варьируя значение порога вхождения, получаем кривую. Данный метод позволяет качественно и количественно сравнивать показатели системы при изменении параметров.

#### 4.1.2 Описание экспериментов

Для работы системы необходимо определить значения ряда параметров, в частности:

- а) количество фраз для стадии обучения модели (см. раздел 2.3). При создании персональной модели источника речи, пользователю необходимо произнести ключевую фразу несколько раз, чтобы накопить достаточное для

обучения модели количество речевых данных. Эксперимент позволит определить такое минимальное количество, при котором для тестируемой выборки будут соблюдены требования, предъявляемые к точности аутентификации;

- б) величина порога вхождения (реализованный алгоритм голосовой верификации даёт численную оценку правдоподобия проверяемой гипотезы, но для принятия решения необходимо определить, достаточна ли полученная величина для того, чтобы с уверенностью говорить о положительном результате аутентификации). От данной величины напрямую зависит чувствительность системы, поэтому, если она выбирается глобально, необходимо получить её экспериментально, то есть найти минимальное значение, при котором исключаются ошибки второго рода (ложные срабатывания).
- в) влияние применения алгоритма удаления тишины на точность аутентификации (алгоритм рассмотрен в разделе 2.7). Целью эксперимента является определение параметров алгоритма, при которых достигается наибольшая точность (в рамках имеющейся тестовой выборки).
- г) количество компонент смеси Гауссиан. Применяемый метод для моделирования источника речи параметризуется количеством компонент смеси. Чем больше это количество, тем, теоретически, точнее ожидаемая точность аутентификации. Эксперимент позволит определить минимальное количество, при котором удовлетворяются требования точности.

### **4.1.3 Описание входных данных**

Для проведения экспериментов была составлена выборка, состоящая из 30 человек, из них 16 мужчин и 15 женщин. Каждый человек записывал одну и ту же фразу, что позволяет анализировать способность системы различать особенности речи отдельных людей. Запись проходила в обстановке, максимально приближенной к условиям эксплуатации программного комплекса: каждый источник речи использовал для записи собственный микрофон и записывался в домашних условиях, удаленно. Средняя длина записанной фразы: 2.913 секунд. Количество записей в расчете на человека: 60.

При определении количества смеси Гауссиан, для каждого источника речи в тестовой выборке производилось обучение моделей из  $k$  по 10 фразам (фразы для обучения). По данным предварительных испытаний, при  $k > 28$  и количестве тестовых данных менее 1 минуты обучение модели заканчивалось неудачей из-за ошибок округления, поэтому  $k = \overline{1,28}$ . При определении необходимого количества фраз для обучения для каждого источника речи производилось обучение моделей по  $n$  фразам, где  $n = \overline{5,20}$ . Количество компонент выбиралось на основе результатов предыдущего эксперимента. При изучении зависимости параметра алгоритма удаления тишины на точность аутентификации производилось обучение моделей из  $k$  компонент по  $n$  фразам, при этом входные данные обрабатывались алгоритмом удаления тишины с различными значениями параметра  $\omega$ :  $\omega \in \{0, 0.05, \dots, 0.65, 0.7\}$ .

Для каждой построенной модели строились показания ошибок I-ого и II-ого рода при значениях порога вхождения  $\Theta \in \{-1500, -1450, -1400, \dots, 1450, 1500\}$ . После вычисления данных показаний, результаты для всех источников речи и одинаковых факторов ( $k, n, \omega$ ) усреднялись для получения средних показаний. По полученным показаниям находилось значение частоты равнозначной ошибки.

При вычислении вероятности ошибки I-ого рода использовались те записи источника речи, которые не были использованы при обучении.

При вычислении вероятности ошибки II-ого рода использовались записи всех источников речи в рамках выбранной субпопуляции.

## Сбор данных

Для сбора данных в программный комплекс был включен модуль, использующий разработанную базу для записи и отправки голосовых данных от анонимных пользователей и сохранения их в файловой системе с кратким описанием, что позволило автоматизировать последующую обработку. Запись данных при этом происходила удаленно. Форма для записи и отправки голосовых данных представлена на рисунке 4.1. Процесс записи аналогичен процессу создания персональной голосовой модели, описанному в разделе 3.3.3.1. После записи, участнику тестирования необходимо было заполнить форму с описанием сессии, которая сохранялась в базе данных для последующей обработки в процессе эксперимента.

The image shows a two-part graphical user interface. The top part, titled 'Загрузка голосовых данных' (Loading voice data), contains instructions: 'Нажмите кнопку «Запись», произнесите ключевую фразу, и нажмите на кнопку «Стоп».' (Click the 'Record' button, speak the key phrase, and click the 'Stop' button.) and a 'Запись' (Record) button. The bottom part, titled 'Подтверждение' (Confirmation), contains a 'Пол:' (Gender) dropdown menu with 'Мужчина' (Male) selected, a 'Фраза:' (Phrase) text input field, an 'Описание:' (Description) text area, and a 'Готово' (Ready) button.

Рисунок 4.1 — Диалог записи и загрузки голосовых данных

## 4.2 Анализ результатов

На рисунке 4.2 отображена зависимость частоты равнозначной ошибки (значение вероятностей ошибок I-ого и II-ого рода, при таком пороге вхождения, при котором эти значения совпадают) от количества компонент в модели. Анализ данной зависимости позволяет заключить, что:

- аутентификация источников речи из мужской субпопуляции происходит с большей точностью;
- при значении количества компонент, равному 20 наблюдается падение точности для обеих субпопуляций;
- для мужской субпопуляции модель, состоящая из 25 Гауссиан позволяет получить наилучший результат по тестовой выборке;
- для женской субпопуляции модель, состоящая из 21 Гауссиан позволяет получить наилучший результат по тестовой выборке.

На рисунке 4.3 отображена зависимость частоты равнозначной ошибки от количества фраз, используемых при обучении модели. Анализ данной зависимости позволяет сделать следующий вывод:

- для мужской субпопуляции точность аутентификации достигает наибольшего значения при сессии регистрации, состоящей из 11 фраз;
- для женской субпопуляции точность аутентификации достигает наибольшего значения при сессии регистрации, состоящей из 14 фраз;



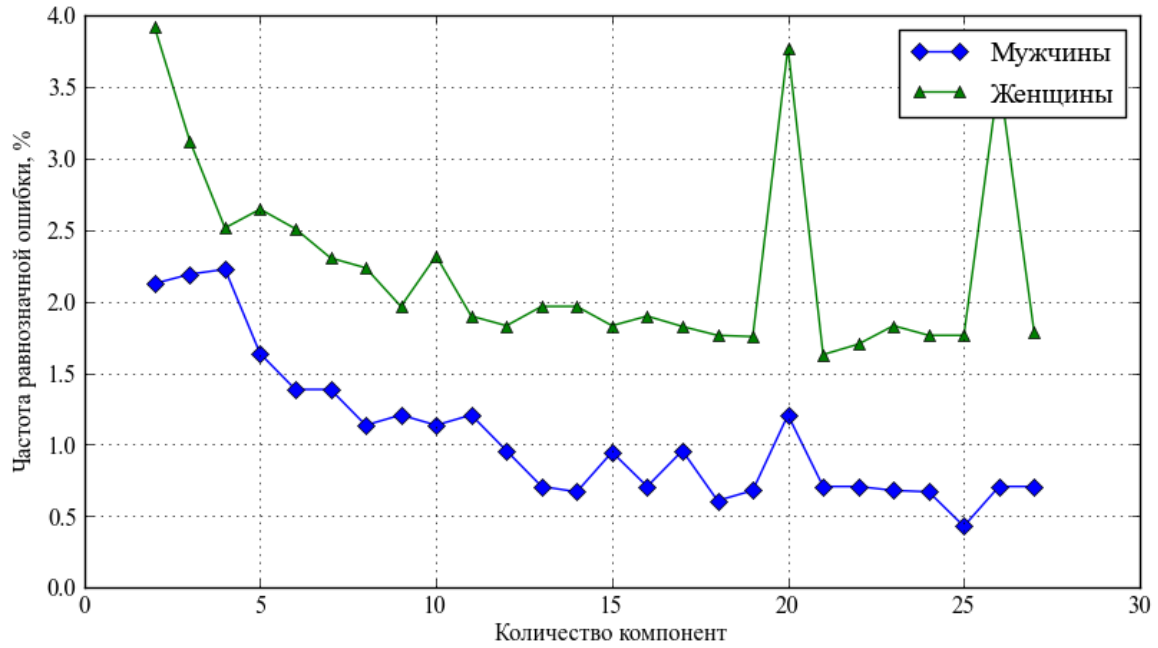


Рисунок 4.2 — Зависимость частоты равнозначной ошибки от количества компонент в смеси Гауссиан

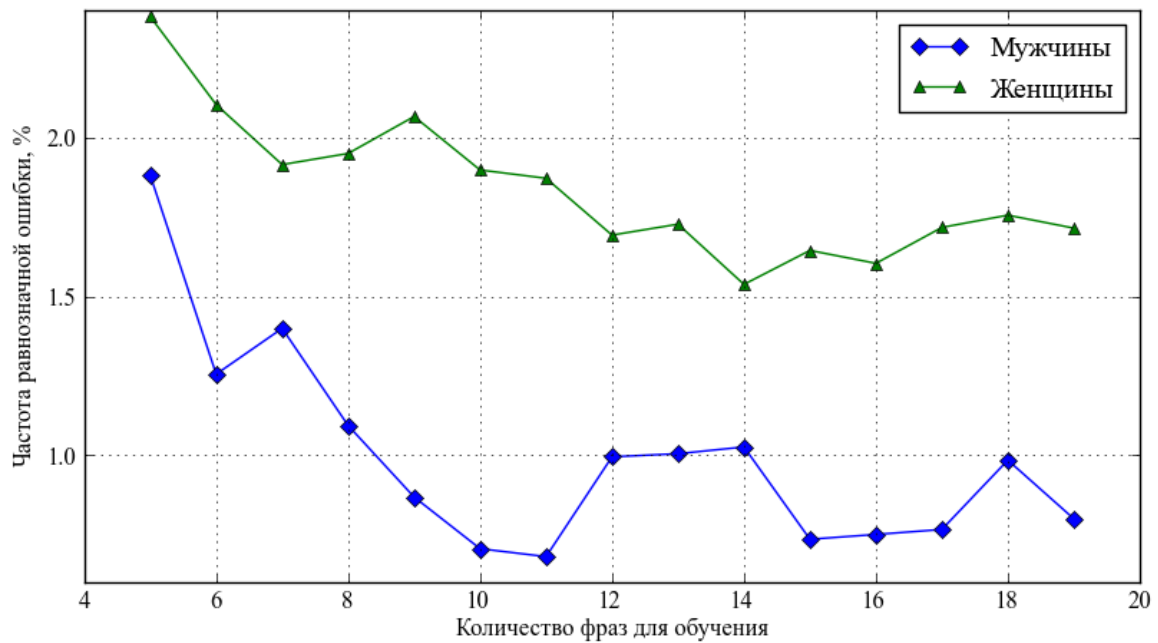


Рисунок 4.3 — Зависимость частоты равнозначной ошибки от количества фраз, записанных при регистрации

На рисунке 4.4 отображена зависимость частоты равнозначной ошибки от значения параметра  $\omega$  алгоритма удаления фрагментов записи, не содержащих

речь. Параметр  $\omega$  (см. раздел 2.7) влияет на чувствительность алгоритма: при  $\omega = 0$  алгоритм не изменяет сигнал, с увеличением  $\omega$  увеличивается количество сегментов, которые алгоритм воспринимает как «тишину». Анализ данной зависимости позволяет сделать вывод о том, что при значении  $\omega > 0.3$  точность системы начинает понижаться, максимальная точность в рамках тестовой выборки достигается при  $\omega = 0.25$  и равна 0.234%.

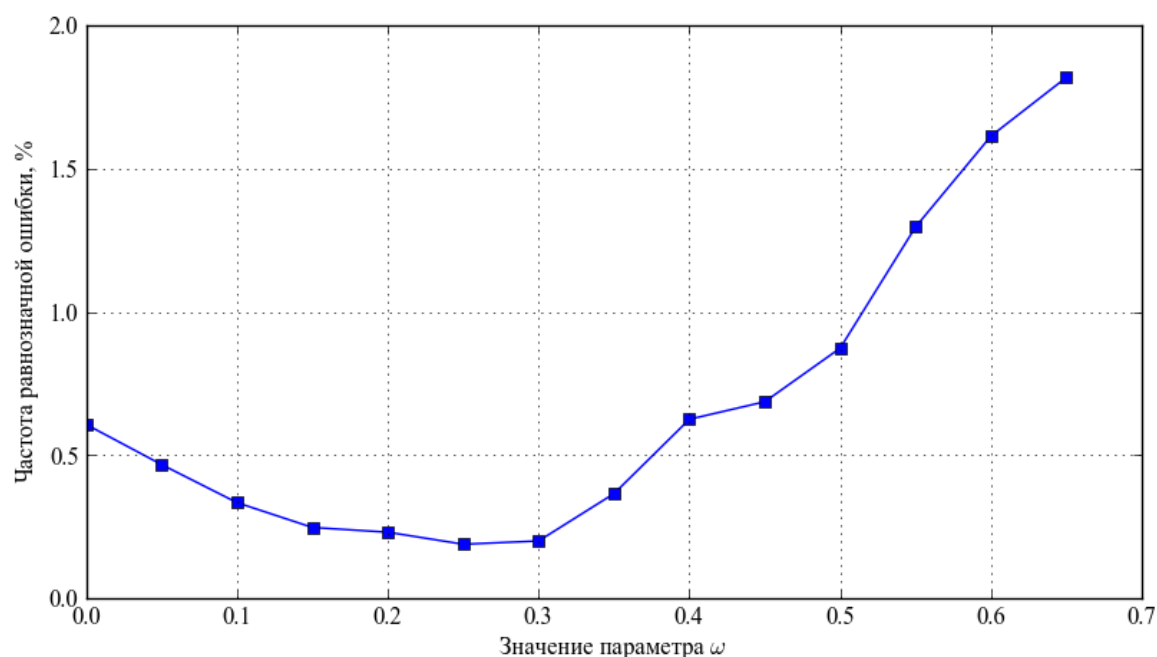


Рисунок 4.4 — Зависимость частоты равнозначной ошибки от значения параметра  $\omega$

На рисунке 4.5 представлена зависимость вероятности ошибок I-ого и II-ого рода от значения порога вхождения при использовании вычисленных ранее параметров. Результаты обеих популяций были скомбинированы. На пересечении кривых находится точка равнозначной ошибки (вероятность при этом равна 0.234%). Как видно из графика, в точке равнозначной ошибки величина порога отрицательна. При этом, если установить допустимое значение вероятности ложных отказов в 20% (вероятность отказа в аутентификации аутентичному источнику), то это позволит установить порог вхождения в 1300 единиц. В данном случае ошибка второго рода оказывается равной нулю (для тестируемой выборки).

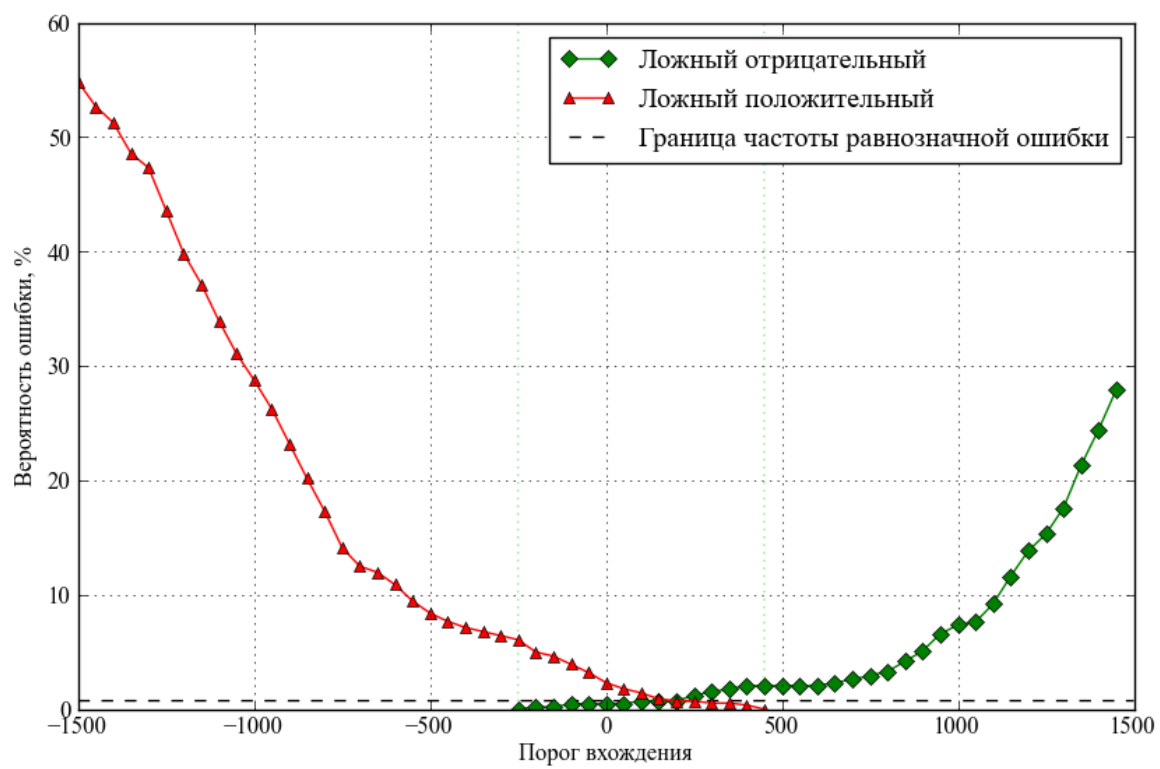


Рисунок 4.5 — Зависимость ошибок аутентификации от значения порога вхождения

## 5 Раздел по охране труда и экологии

### 5.1 Анализ опасных и вредных факторов при разработке программного обеспечения и мероприятия по их устранению

Разработка программного обеспечения требует длительного взаимодействия с вычислительными системами. Работа с персональными электронно-вычислительными машинами связана с рядом вредных и опасных факторов, таких как статическое электричество, рентгеновское излучение, электромагнитные поля, блики и отраженный свет, ультрафиолетовое излучение, мерцание изображения. При длительном воздействии на организм эти факторы негативно влияют на здоровье человека.

#### 5.1.1 Микроклимат

Работа как программиста, так и пользователя относится к категории 1а, поскольку не предполагает больших физических усилий. Поэтому оптимальные нормы микроклимата для рабочего помещения программиста определяются табл. 5.1 (СанПиН 2.2.2/2.4.1340-03).

Таблица 5.1 — Оптимальные нормы микроклимата

|          | Температура<br>воздуха, °С | Относительная<br>влажность воздуха, % | Скорость<br>движения<br>воздуха, м/с |
|----------|----------------------------|---------------------------------------|--------------------------------------|
| Холодный | 22-24                      | 40-60                                 | 0.1                                  |
| Теплый   | 23-25                      | 40-60                                 | 0.1                                  |

Вредным фактором при работе с ЭВМ является также запыленность помещения. Этот фактор усугубляется влиянием на частицы пыли электростатических полей персональных компьютеров.

Для устранения несоответствия параметров указанным нормам проектом предусмотрено использование системы кондиционирования как наиболее эффективного и автоматически функционирующего средства.

Нормы СанПиН 2.2.4.1294-03 «Санитарно-гигиенические нормы допустимых уровней ионизации воздуха» определяют уровни положительных и отрицательных ионов в воздухе (табл. 5.2):

Таблица 5.2 — Уровни ионизации воздуха помещений при работе на ВДТ и ПЭВМ

| Уровни                 | Число ионов в 1 см <sup>3</sup> воздуха |           |
|------------------------|---|-----------|
|                        | $n^+$                                   | $n^-$     |
| Минимально необходимые | 400                                     | 600       |
| Оптимальные            | 1500-3000                               | 3000-5000 |
| Предельно допустимые   | 50000                                   | 50000     |

Для обеспечения требуемых уровней предусмотрено использование системы ионизации Сапфир-4А.

Концентрация вредных химических веществ в помещениях с ПЭВМ не должна превышать «ПДК загрязняющих веществ в атмосферном воздухе населенных мест» ГН 2.1.6.789-99. Для выполнения указанных требований предусмотрено применение фильтров из активированного угля.

### 5.1.2 Шум и вибрации

Уровень шума на рабочем месте программиста не должен превышать 50 дБА, а уровень вибрации не должен превышать допустимых норм вибрации. СанПиН 2.2.2.542-96 устанавливает следующие нормы на вибрацию (табл. 5.3).

Таблица 5.3 — Допустимые нормы вибрации на рабочих местах с ВДТ и ПЭВМ

| Среднегеометрические частоты октавных полос, Гц | Допустимые значения по виброскорости |    |
|---|--------------------------------------|----|
|   | м/с                                  | дБ |
| 2   | $4.5 \times 10$                      | 79 |
| 4   | $2.2 \times 10$                      | 73 |
| 8   | $1.1 \times 10$                      | 67 |
| 16  | $1.1 \times 10$                      | 67 |
| 31.5  | $1.1 \times 10$                      | 67 |
| 63  | $1.1 \times 10$                      | 67 |
| Корректированные значения и их уровни в дБ      | $2.0 \times 10$                      | 72 |

При разработке программного обеспечения внутренними источниками шума являются вентиляторы, а также принтеры и другие периферийные устройства ЭВМ.

Внешние источники шума – прежде всего, шум с улицы и из соседних помещений. Постоянные внешние источники шума, превышающего нормы, отсутствуют.

Для устранения превышения нормы проектом предусмотрено применение звукопоглощающих материалов для облицовки стен и потолка помещения, в котором осуществляется работа с вычислительной техникой.

### 5.1.3 Освещение

Наиболее важным условием эффективной работы программистов и пользователей является соблюдение оптимальных параметров системы освещения в рабочих помещениях.

Естественное освещение осуществляется через светопроемы, ориентированные в основном на север и северо-восток (для исключения попадания прямых солнечных лучей на экраны компьютеров) и обеспечивает коэффициент естественной освещенности (КЕО) не ниже 1.5%.

В качестве искусственного освещения проектом предусмотрено использование системы общего равномерного освещения. В соответствии с СанПиН 2.2.2/2.4.1340-03 освещенность на поверхности рабочего стола находится в пределах 300-500 лк. Разрешается использование светильников местного освещения для работы с документами (при этом светильники не должны создавать блики на поверхности экрана).

Правильное расположение рабочих мест относительно источников освещения, отсутствие зеркальных поверхностей и использование матовых материалов ограничивает прямую (от источников освещения) и отраженную (от рабочих поверхностей) блескость. При этом яркость светящихся поверхностей не превышает  $200 \frac{\text{кд}}{\text{м}^2}$ , яркость бликов на экране ПЭВМ не превышает  $40 \frac{\text{кд}}{\text{м}^2}$  и яркость потолка не превышает  $200 \frac{\text{кд}}{\text{м}^2}$ .

В соответствии с СанПиН 2.2.2/2.4.1340-03 проектом предусмотрено использование люминесцентных ламп типа ЛБ в качестве источников света при

искусственном освещении. В светильниках местного освещения допускается применение ламп накаливания.

Применение газоразрядных ламп в светильниках общего и местного освещения обеспечивает коэффициент пульсации не более 5%.

Таким образом, проектом обеспечиваются оптимальные условия освещения рабочего помещения.

#### 5.1.4 Рентгеновское излучение

В соответствии с СанПиН 2.2.2/2.4.1340-03, проектом предусмотрено использование ПЭВМ, конструкция которых обеспечивает мощность экспозиционной дозы рентгеновского излучения в любой точке на расстоянии 0.05 м от экрана и корпуса монитора не более  $0.1 \frac{\text{мбэр}}{\text{ч}}$  ( $100 \frac{\text{мкР}}{\text{ч}}$ ). Результаты сравнения норм излучения приведены в табл. 5.4.

Таблица 5.4 — Сравнение норм рентгеновского излучения в различных стандартах

|                          | Допустимое значение,<br>$\frac{\text{мкР}}{\text{ч}}$ не более |
|--------------------------|--|
| СанПиН 2.2.2/2.4.1340-03 | 100  |
| ТСО-99                   | 500  |
| MPR II                   | 500  |

Как видно из табл. 5.4, стандарты MPR II и ТСО-99 предъявляют менее жесткие требования к рентгеновскому излучению, чем СанПиН. Но при соблюдении оптимального расстояния между пользователем и монитором дозы рентгеновского излучения не опасны для большинства людей.

#### 5.1.5 Неионизирующие электромагнитные излучения

В соответствии с СанПиН 2.2.2/2.4.1340-03, допустимые значения параметров неионизирующих излучений приводятся в табл. 5.5 и табл. 5.6.

Величина поверхностного электростатического потенциала не должна превышать 500 В.

Таблица 5.5 — Предельно допустимые значения напряженности электрического

| поля            |                                    |
|-----------------|------------------------------------|
| Диапазон частот | Допустимые значения, $\frac{В}{м}$ |
| 5 Гц - 2 кГц    | 25                                 |
| 2 кГц - 400 кГц | 2.5                                |

Таблица 5.6 — Предельно допустимые значения магнитной индукции

| Диапазон частот | Допустимые значения, нТл |
|-----------------|--------------------------|
| 5 Гц - 2 кГц    | 250                      |
| 2 кГц - 400 кГц | 25                       |

Мониторы, используемые в настоящее время, удовлетворяют нормам МРР II (или более жестким требованиям) и имеют предельные значения, определенные в табл. 5.7 и табл. 5.8.

Таблица 5.7 — Предельно допустимые значения напряженности электрического

| поля            |                                    |
|-----------------|------------------------------------|
| Диапазон частот | Допустимые значения, $\frac{В}{м}$ |
| 5 Гц - 2 кГц    | 25                                 |
| 2 кГц - 400 кГц | 2.5                                |

Поверхностный электростатический потенциал не превышает 500 В.

Таким образом, параметры электрических и магнитных (неионизирующих) полей удовлетворяют требованиям СанПиН.

### 5.1.6 Визуальные параметры

Неправильный выбор визуальных эргономических параметров приводит к ухудшению здоровья пользователей, быстрой утомляемости, раздражительности. В этой связи, проектом предусмотрено, что конструкция вычислительной системы и ее эргономические параметры обеспечивают комфортное и надежное считывание информации.

Требования к визуальным параметрам, их внешнему виду, дизайну, возможности настройки представлены в СанПиН 2.2.2/2.4.1340-03. Визуальные эргономические параметры монитора и пределы их изменений приведены в табл. 5.9.



Таблица 5.8 — Предельно допустимые значения магнитной индукции

| Диапазон частот | Допустимые значения, нТл |
|-----------------|--------------------------|
| 5 Гц - 2 кГц    | 200                      |
| 2 кГц - 400 кГц | 25                       |

Таблица 5.9 — Визуальные эргономические параметры ВДТ и пределы их изменений

| Наименование параметров  | Пределы значений параметров |                     |
|--|-----------------------------|---------------------|
|  | мин.<br>(не менее)          | макс.<br>(не более) |
| Яркость знака (яркость фона), $\frac{\text{кД}}{\text{м}^2}$ ,<br>измеренная в темноте | 35                          | 120                 |
| Внешняя освещенность экрана, лк  | 100                         | 250                 |
| Угловой размер знака, угл. мин.  | 16                          | 60                  |

Для выполнения этих требований проектом предусмотрено использование современных мониторов, имеющих достаточно широкий набор регулируемых параметров. В частности, для удобного считывания информации реализована возможность настройки положения монитора по горизонтали и вертикали. Мониторы оснащены специальными устройствами и средствами настройки ширины, высоты, яркости, контраста и разрешения изображения. Кроме того, в современных мониторах зерно изображения имеет размер в пределах 0.27 мм, что обеспечивает высокую четкость и непрерывность изображения. Наконец, на поверхность дисплея нанесено матовое покрытие, чтобы избавиться от солнечных бликов.

## 5.2 Расчет системы искусственного освещения

В зависимости от цели расчета при проектировании искусственного освещения приходится решать следующий ряд вопросов:

- а) Выбрать или определить типы ламп и светильников. Для освещения предприятий службы быта следует применять газоразрядные лампы. Применение ламп накаливания целесообразно при температуре воздуха ниже 10 °С и падении напряжения в сети более 10% от номинального. Выбор светильника должен производиться с учетом его крепления, подвода электроэнергии, защиты от механических повреждений, взрыво- и пожароопасности (откры-

тые, закрытые, пылевлагонепроницаемые, взрывоопасные, взрывозащищенные светильники).

- б) Выбрать систему освещения. Наиболее экономичной является система комбинированного освещения, так как она создает наиболее равномерное светораспределение. При комбинированном освещении доля общего освещения в нем не должна быть меньше 10%.
- в) Выбрать расположение светильников и определить их количество. Светильники, расположенные симметрично вдоль или поперек помещения, в шахматном порядке, рядами, ромбовидно, обеспечивают равномерное по площади освещение. Локализованное неравномерное размещение светильников производят с учетом местонахождения ПЭВМ, оборудования и т.д. Экспериментально установлено, что наибольшая равномерность достигается:

- При шахматном расположении, если  $\frac{r}{H_p} \leq 1.7 \div 2.5$
- При расположении прямоугольником, если  $\frac{r}{H_p} \leq 1.4 \div 2.0$

где  $r$  - расстояние между светильниками, м;

$H_p$  - высота подвеса светильника над рабочей поверхностью, м:

$$H_p = H - h_c - h_{\text{р. м.}} \quad (5.1)$$

где  $H$  - высота помещения, м;

$h_c$  - высота подвеса светильника, м;

$h_{\text{р. м.}}$  - высота рабочего места ( $h_{\text{р. м.}} = 0.8$  м), м.

Оптимальное расстояние от крайнего ряда светильников до стены:  $r_k = (0.24 \div 0.3)r$ .

При отсутствии рабочих поверхностей у стены:  $r_k = (0.4 \div 0.5)r$ . Для исключения слепящего действия светильников общего освещения должно выполняться правило  $H - h_c \leq 2.5 \div 4$  м при мощности ламп  $P_{\text{л}} \leq 200$  Вт. Необходимое число светильников при расположении квадратом составляет:

$$N_c = \frac{S}{r^2} \quad (5.2)$$

где  $S$  – площадь помещения,  $\text{м}^2$ ;

$r$  – длина стороны квадрата,  $\text{м}$ .

- г) Определить нормируемую освещенность рабочего места по минимальному размеру объекта различия, фону, контрасту объекта с фоном в системе освещения.

Для расчета искусственного освещения используют три метода:

- Метод светового потока для общего равномерного освещения горизонтальной рабочей поверхности.
- Точечный метод для любой системы освещения.
- Метод удельной мощности для ориентировочных расчетов общего равномерного освещения.

Световой поток определяется по формуле:

$$F_{\text{л}} = \frac{E_{\text{н}} \cdot K \cdot S \cdot Z}{N \cdot \eta} \quad (5.3)$$

где  $F_{\text{л}}$  – световой поток лампы, лк;

$E_{\text{н}}$  – нормированная освещенность, лк;

$S$  – площадь освещаемого помещения,  $\text{м}^2$ ;

$K$  – коэффициент запаса (в соответствии со СНиП 23-05-95 для люминесцентных ламп производственных цехов предприятий службы быта  $K = 1.6 \div 1.7$ ; для остальных помещений  $K = 1.5$ );

$Z$  – коэффициент минимальной освещенности, равный отношению средней освещенности к минимальной;

$N$  – число ламп;

$\eta$  – коэффициент использования светового потока, равный отношению потока, падающего на рабочую поверхность, к общему потоку ламп.

Коэффициент использования светового потока  $\eta$  зависит от к.п.д. светильника, коэффициента отражения потолка ( $\rho_{\text{п}}$ ), стен ( $\rho_{\text{с}}$ ), величины показателя помещения  $i$ , учитывающего геометрические параметры помещения, высоту подвеса светильника ( $H_{\text{п}}$ ):

$$i = \frac{a \cdot b}{H_{\text{п}}(a + b)} \quad (5.4)$$

где  $a$  и  $b$  – ширина и длина помещения, м.

Исходные данные:

- длина рабочего помещения  $a = 16$  м, ширина  $b = 10$  м, высота  $H = 3.6$  м
- $E_{\text{н}} = 400$  лк
- $F_{\text{л}} = 5220$  лк - световой поток одной лампы ЛБ-80.

Расчет: По формулам 5.4 и 5.1  $i = \frac{a \cdot b}{(H - h_c - h_{\text{п. м.}})(a + b)} = \frac{10 \cdot 16}{(3.6 - 0.1 - 0.8)(10 + 16)} \approx 2.3$ .

Исходя из этого  $\eta = 0.41$

Исходя из формулы 5.3 общее число ламп  $N = \frac{400 \cdot 160 \cdot 1.6 \cdot 1.1}{5220 \cdot 0.41} = 52$ . При использовании светильников с двумя лампами потребуется 18 светильников.

Расстояние между двумя светильниками составит  $r = 1.5 \cdot (3.6 - 0.1 - 0.8) = 4$  м.

Расстояние от стены до светильников  $r_k = 0.25 \cdot 4 = 1$  м.

Светильники следует расположить в 3 ряда по 6 светильников в каждом (рис. 5.1) Суммарная потребляемая мощность светильников составляет  $P_{\Sigma} = 18 \cdot 3 \cdot 80 = 4320$  Вт.

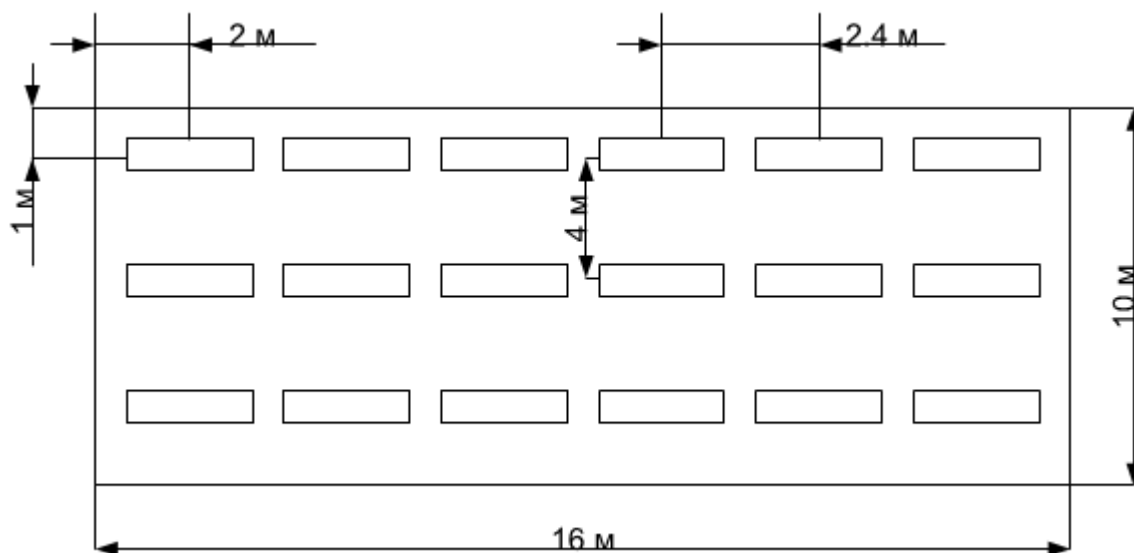


Рисунок 5.1 — Схема освещения помещения

При этом имеет место избыточное освещение, превышающее расчетный световой поток на  $\frac{54-52}{52} = 3.8\%$  (допустимым является 20% отклонение). Таким образом, в проекте используются 18 светильников с высотой подвеса 0.1 м и, со-

ответственно, 54 люминесцентных лампы ЛБ-80 со световым потоком 5220 лм и световой отдачей  $65.3 \frac{\text{лм}}{\text{Вт}}$ . Потребляемая мощность от светильников составляет 4320 Вт.

## 6 Технико-экономический раздел

### 6.1 Организация и планирование процесса разработки

Организационно-экономическая часть процесса разработки программного продукта предусматривает выполнение следующих работ:

- формирование состава выполняемых работ и группировка их по стадиям разработки;
- расчет трудоемкости проекта;
- определение продолжительности выполнения отдельных этапов разработки;
- календарный график выполнения проекта;
- контроль выполнения календарного графика.

#### 6.1.1 Формирование состава выполняемых работ и группировка их по стадиям разработки

Разработку программного продукта можно разделить на следующие стадии:

Таблица 6.1: Состав выполняемых работ и стадии разработки программного продукта

| Стадии разработки программного продукта | Состав работ, выполняемых разработчиками постановки задачи   | Состав работ, выполняемых разработчиками программного продукта                             |
|---|--|--|
| Техническое задание (ТЗ)                | Постановка задач, выбор критериев эффективности. Разработка технико-экономического обоснования разработки. Выбор средств программирования. Предварительный выбор методов выполнения работы. Разработка календарного плана. | Определение состава пакета прикладных программ, состава и структуры.                       |
| Эскизный проект (ЭП)                    | Предварительная разработка структуры входных и выходных данных. Разработка общего описания алгоритмов реализации решения задач. Разработка пояснительной записки. Согласование и утверждение эскизного проекта.            | Консультация разработчиков постановки задач. Согласование и утверждение эскизного проекта. |

Продолжение табл. 6.1

|                         |   |   |
|-------------------------|---|---|
| Технический проект (ТП) | Разработка алгоритмов решения задач. Разработка пояснительной записки. Согласование и утверждение технического проекта. Уточнение структуры, анализ и определение формы предоставления входных и выходных данных. Выбор конфигурации технических средств.                                     | Разработка структуры программы. Разработка программной документации и передача ее для включения в технический проект. Уточнение структуры, анализ и определение формы предоставления входных и выходных данных. Выбор конфигурации технических средств. |
| Рабочий проект (РП)     | Комплексная отладка задач и сдача в опытную эксплуатацию. Разработка проектной документации. Разработка, согласование программы и методики испытаний. Предварительное проведение испытаний.   | Программирование и отладка программ. Описание контрольного примера. Разработка программной документации. Разработка, согласование программы и методики испытаний. Предварительное проведение всех испытаний.  |
| Внедрение (В)           | Подготовка и передача программной документации для сопровождения с оформлением соответствующего Акта. Передача программной продукции в фонд алгоритмов и программ. Проверка алгоритмов и программ решения задач, корректировка документации после опытной эксплуатации программного продукта. | Проверка алгоритмов и программ решения задач, корректировка документации после опытной эксплуатации программного продукта.  |

Планирование длительности этапов и содержания проекта осуществляется в соответствии с ЕСПД ГОСТ 34.603-92 и распределяет работы по этапам, как показано в табл. 6.2.

### 6.1.2 Расчет трудоемкости выполнения работ

Трудоемкость разработки программной продукции зависит от ряда факторов, основными из которых являются следующие:

- степень новизны разрабатываемого программного комплекса;

Таблица 6.2 — Распределение работ проекта по этапам

| Этап | Основные стадии      | №  | Содержание работы  |
|------|----------------------|----|--|
| 1    | Техническое задание  | 1  | Постановка задачи  |
|      |                      | 2  | Анализ предметной области  |
|      |                      | 3  | Выбор средств разработки, сторонних библиотек                    |
| 2    | Эскизный проект      | 4  | Разработка библиотеки голосовой аутентификации                   |
|      |                      | 5  | Разработка схемы хранения данных                                 |
|      |                      | 6  | Разработка пользовательского интерфейса                          |
|      |                      | 7  | Разработка контроллера   |
| 3    | Техно-рабочий проект | 8  | Реализация алгоритмов голосовой аутентификации                   |
|      |                      | 9  | Реализация пользовательского интерфейса                          |
|      |                      | 10 | Реализация контроллера   |
|      |                      | 11 | Отладка программного комплекса                                   |
|      |                      | 12 | Экспериментальное определение параметров и характеристик системы |
|      |                      | 13 | Разработка документации к системе                                |
|      |                      | 14 | Итоговое тестирование системы                                    |
| 4    | Внедрение            | 15 | Установка и настройка программного продукта                      |

- сложность алгоритма его функционирования;
- объем используемой информации, вид ее представления и способ обработки;
- уровень используемого алгоритмического языка программирования (Чем выше уровень языка, тем меньше трудоемкость).

Разрабатываемый программный продукт можно отнести:

- По степени новизны — к группе В. Разработка программной продукции, не имеющей аналогов.
- По степени сложности алгоритма функционирования — к 3-ей группе. Разработка программной продукции, реализующей алгоритмы стандартных методов решения задач.
- По виду представления исходная информация относится к группе 12 (исходная информация представлена в форме документов, имеющих одинаковый формат и структуру, требуется форматный контроль информации).
- Структура выходных документов относится к группе 22 (требуется вывод на печать одинаковых документов, вывод информационных массивов на машинные носители).
- Количество разновидностей форм входной информации - 3.



- Количество разновидностей форм выходной информации - 3.

Трудоемкость разработки программной продукции  $\tau_{ПП}$  может быть определена как сумма величин трудоемкости выполнения отдельных стадий разработки ПП из выражения 6.1.

$$\tau_{ПП} = \tau_{ТЗ} + \tau_{ЭП} + \tau_{РП} + \tau_{В} \quad (6.1)$$

где  $\tau_{ТЗ}$  — трудоемкость разработки технического задания на создание ПП;  
 $\tau_{ЭП}$  — трудоемкость разработки эскизного проекта ПП;  
 $\tau_{ТП}$  — трудоемкость разработки технического проекта ПП;  
 $\tau_{РП}$  — трудоемкость разработки рабочего проекта ПП;  
 $\tau_{В}$  — трудоемкость внедрения разработанного ПП.

Трудоемкость разработки технического задания рассчитывается по формуле 6.2:

$$\tau_{ТЗ} = T_{РЗ}^3 + T_{РП}^3 \quad (6.2)$$

где  $T_{РЗ}^3$  - затраты времени разработчика постановки задач на разработку ТЗ, чел.-дни;  
 $T_{РП}^3$  - затраты времени разработчика программного обеспечения на разработку ТЗ, чел.-дни.

Значения величин  $T_{РЗ}^3$  и  $T_{РП}^3$  рассчитываются по формулам 6.3, 6.4:

$$T_{РЗ}^3 = t_3 \cdot K_{РЗ}^3 \quad (6.3)$$

$$T_{РП}^3 = t_3 \cdot K_{РП}^3 \quad (6.4)$$

где  $t_3$  — норма времени на разработку ТЗ на программный продукт в зависимости от функционального назначения и степени новизны разрабатываемого ПП, чел.-дни;

$K_{РЗ}^3$  — коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки на стадии ТЗ;

$K_{РП}^3$  — коэффициент, учитывающий удельный вес трудоемкости работ, выполняемых разработчиком постановки на стадии РП.

$$t_3 = 36 \text{ чел.-дн. (задачи расчетного характера)}$$

$$K_{P3}^3 = 0.65 \text{ (совместная разработка)}$$

$$K_{P\Pi}^3 = 0.35 \text{ (совместная разработка)}$$

$$\tau_{T3} = 36 \cdot (0.65 + 0.35) = 36 \text{ чел.-дн.}$$

Аналогично, по формуле 6.5, рассчитывается трудоемкость эскизного проекта ПП  $\tau_{ЭП}$ .

$$\tau_{ЭП} = T_{P3}^3 + T_{P\Pi}^3 \quad (6.5)$$

$$t_3 = 101 \text{ чел.-дн. (задачи расчетного характера)}$$

$$K_{P3}^3 = 0.65 \text{ (совместная разработка)}$$

$$K_{P\Pi}^3 = 0.35 \text{ (совместная разработка)}$$

$$\tau_{T3} = 101 \cdot (0.65 + 0.35) = 101 \text{ чел.-дн.}$$

Трудоемкость разработки технического проекта  $\tau_{ТП}$  зависит от функционального назначения программного продукта, количества разновидностей форм входной и выходной информации и определяется как сумма времени, затраченного разработчиком постановки задач и разработчиком программного обеспечения. Эти зависимости отражает формула 6.6.

$$\tau_{ТП} = (t_{P3}^T + t_{P\Pi}^T) \cdot K_B \cdot K_P \quad (6.6)$$

где  $t_{P3}^T, t_{P\Pi}^T$  — норма времени, затрачиваемого на разработку ТП разработчиком постановки задач и разработчиком программного обеспечения соответственно, чел.-дни;

$K_B$  — коэффициент учета вида используемой информации;

$K_P$  — коэффициент учета режима обработки информации.

Значение коэффициента  $K_B$  определяется из выражения:

$$K_B = \frac{K_{\Pi} \cdot n_{\Pi} + K_{НС} \cdot n_{НС} + K_B \cdot n_B}{n_{\Pi} + n_{НС} + n_B} \quad (6.7)$$

где  $K_{\Pi}, K_{НС}, K_B$  — значения коэффициентов учета вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно;

$n_{\Pi}, n_{НС}, n_B$  — количество наборов данных переменной, нормативно-справочной информации и баз данных соответственно.

$$K_{\Pi} = 1.2, K_{НС} = 1.08, K_B = 3.12$$

$$n_{\Pi} = 6, n_{\text{НС}} = 4, K_{\text{Б}} = 0$$

По формуле 6.7:

$$K_{\text{В}} = 1.15$$

$$K_{\text{Р}} = 1.45$$

По формуле 6.6:

$$\tau_{\text{ТП}} = (14 + 12) \cdot 1.15 \cdot 1.45 = 43 \text{ чел.-дн.}$$

Трудоемкость разработки рабочего проекта  $\tau_{\text{РП}}$  зависит от функционального назначения ПП, количества разновидностей форм входной и выходной информации, сложности алгоритма функционирования, сложности контроля информации, степени использования готовых программных модулей, уровня алгоритмического языка программирования и определяется по формуле:

$$\tau_{\text{РП}} = K_{\text{К}} \cdot K_{\text{Р}} \cdot K_{\text{Я}} \cdot K_{\text{З}} \cdot K_{\text{ИА}} \cdot (t_{\text{РЗ}}^T + t_{\text{РП}}^T) \quad (6.8)$$

где  $K_{\text{К}}$  — коэффициент учета сложности контроля информации;

$K_{\text{Я}}$  — коэффициент учета уровня используемого алгоритмического языка программирования;

$K_{\text{З}}$  — коэффициент учета степени использования готовых программных модулей;

$K_{\text{ИА}}$  — коэффициент учета вида используемой информации и сложности алгоритма ПП;

$t_{\text{РЗ}}^T, t_{\text{РП}}^T$  — норма времени, затраченного на разработку РП на алгоритмическом языке высокого уровня разработчиком постановки задач и разработчиком программного обеспечения соответственно, чел.-дни.

Значение коэффициента  $K_{\text{ИА}}$  определяется из выражения:

$$K_{\text{ИА}} = \frac{K_{\Pi}' \cdot n_{\Pi} + K_{\text{НС}}' \cdot n_{\text{НС}} + K_{\text{Б}}' \cdot n_{\text{Б}}}{n_{\Pi} + n_{\text{НС}} + n_{\text{Б}}} \quad (6.9)$$

где  $K_{\Pi}'$ ,  $K_{\text{НС}}'$ ,  $K_{\text{Б}}'$  - значения коэффициентов учета сложности алгоритма ПП и вида используемой информации для переменной, нормативно-справочной информации и баз данных соответственно.

$$K_{\text{К}} = 1$$

$$K_{\text{Р}} = 1.52$$

$$K_{\text{Я}} = 1 \text{ (алгоритмический язык высокого уровня)}$$

$$K_{\text{З}} = 0.5 \text{ (использование более 60\% готовых программных модулей)}$$

$$\tau_{\text{РЗ}} = 14, \tau_{\text{РП}} = 12$$

По формуле 6.9:

$$K_{\text{ИА}} = 1.01$$

По формуле 6.8:

$$\tau_{\text{РП}} = 44 \cdot 1 \cdot 1.52 \cdot 1 \cdot 0.5 \cdot 1.01 = 34 \text{ чел.-дн.}$$

Так как при разработке ПП стадии «Технический проект» и «Рабочий проект» объединены в стадию «Техно-рабочий проект», то трудоемкость ее выполнения  $\tau_{\text{РП}}$  определяется по формуле:

$$\tau_{\text{ТРП}} = 0.85 \cdot \tau_{\text{ТП}} + \tau_{\text{РП}} \quad (6.10)$$

Таким образом  $\tau_{\text{ТРП}} = 0.85 \cdot 43 + 34 = 71 \text{ чел.-дн.}$

Трудоемкость выполнения стадии внедрения  $\tau_{\text{В}}$  может быть рассчитана по формуле:

$$\tau_{\text{В}} = (t_{\text{РЗ}}^{\text{В}} + t_{\text{РП}}^{\text{В}}) \cdot K_{\text{К}} \cdot K_{\text{Р}} \cdot K_{\text{З}} \quad (6.11)$$

где  $t_{\text{РЗ}}^{\text{В}}$ ,  $t_{\text{РП}}^{\text{В}}$  — норма времени, затрачиваемого разработчиком постановки задач и разработчиком программного обеспечения соответственно на выполнение процедур внедрения ПП, чел.-дни.

$$t_{\text{РЗ}}^{\text{В}} = 13, t_{\text{РП}}^{\text{В}} = 20$$

По формуле 6.11:

$$\tau_{\text{В}} = (13 + 20) \cdot 1 \cdot 1.52 \cdot 0.5 = 25 \text{ чел.-дн.}$$

Подставив полученные данные в формулу 6.1, получим:

$$\tau_{\text{ПП}} = 36 + 101 + 71 + 25 = 233 \text{ чел.-дн.}$$

### 6.1.3 Расчет количества исполнителей

Средняя численность исполнителей при реализации проекта разработки и внедрения ПО определяется соотношением:

$$N = \frac{Q_{\text{Р}}}{F} \quad (6.12)$$

где  $Q_{\text{Р}}$  — затраты труда на выполнение проекта (разработка и внедрение ПО);

$F$  — фонд рабочего времени.

Величина фонда рабочего времени определяется соотношением:

$$F = T \cdot F_{\text{М}} \quad (6.13)$$

Таблица 6.3 — Распределение работ проекта по этапам

| Этап  | Трудоемкость эта-<br>па, чел.-дн. | Основные стадии      | №  | Содержание работы   | Трудоемкость,<br>чел.-дн. |
|-------|-----------------------------------|----------------------|----|---|---------------------------|
| 1     | 36                                | Техническое задание  | 1  | Постановка задачи   | 10                        |
|       |                                   |                      | 2  | Анализ предметной области   | 15                        |
|       |                                   |                      | 3  | Выбор средств разработки, сторонних библиотек                         | 11                        |
| 2     | 101                               | Эскизный проект      | 4  | Разработка библиотеки голосовой аутентифика-<br>ции                   | 40                        |
|       |                                   |                      | 5  | Разработка схемы хранения данных                                      | 13                        |
|       |                                   |                      | 6  | Разработка пользовательского интерфейса                               | 20                        |
|       |                                   |                      | 7  | Разработка контроллера  | 28                        |
| 3     | 71                                | Техно-рабочий проект | 8  | Реализация алгоритмов голосовой аутентифика-<br>ции                   | 10                        |
|       |                                   |                      | 9  | Реализация пользовательского интерфейса                               | 8                         |
|       |                                   |                      | 10 | Реализация контроллера  | 11                        |
|       |                                   |                      | 11 | Отладка программного комплекса  | 11                        |
|       |                                   |                      | 12 | Экспериментальное определение параметров и ха-<br>рактеристик системы | 12                        |
|       |                                   |                      | 13 | Разработка документации к системе                                     | 10                        |
|       |                                   |                      | 14 | Итоговое тестирование системы   | 9                         |
| 4     | 25                                | Внедрение            | 15 | Установка и настройка программного продукта                           | 25                        |
| Всего | 233                               |                      |    |   |                           |

где  $T$  — время выполнения проекта в месяцах, равное 5 месяцам;

$F_M$  — фонд времени в текущем месяце, который рассчитывается из учета общества числа дней в году, числа выходных и праздничных дней:

$$F_M = \frac{t_p \cdot (D_K - D_B - D_{\Pi})}{12} \quad (6.14)$$

где  $t_p$  — продолжительность рабочего дня;

$D_K$  — общее число дней в году;

$D_B$  — число выходных дней в году;

$D_{\Pi}$  — число праздничных дней в году.

По формуле 6.14:

$$F_M = \frac{8 \cdot (365 - 103 - 11)}{12} = 162$$

По формуле 6.13:

$$F = 5 \cdot 162 = 810$$

По формуле 6.12 получим число исполнителей проекта:

$$N = \frac{233 \cdot 8}{810} = 3$$

#### 6.1.4 Календарный план-график разработки

Планирование и контроль хода выполнения разработки проводится по календарному графику выполнения работ.

Таблица 6.4 — Планирование разработки

| Стадия разработки   | Трудоемкость | Должность исполнителя | Распределение трудоемкости | Численность |
|---------------------|--------------|-----------------------|----------------------------|-------------|
| Техническое задание | 36           | Ведущий программист   | 20 (56%)                   | 1           |
|                     |              | Программист 1         | 8                          | 1           |
|                     |              | Программист 2         | 8                          | 1           |
| Эскизный проект     | 101          | Ведущий программист   | 51 (51%)                   | 1           |
|                     |              | Программист 1         | 25                         | 1           |
|                     |              | Программист 2         | 25                         | 1           |
| Технический проект  | 71           | Ведущий программист   | 23 (53%)                   | 1           |
|                     |              | Программист 1         | 10                         | 1           |
|                     |              | Программист 2         | 10                         | 1           |
| Внедрение           | 25           | Ведущий программист   | 13 (52%)                   | 1           |
|                     |              | Программист 1         | 6                          | 1           |
|                     |              | Программист 2         | 6                          | 1           |
| Итого               | 233          |                       |                            |             |

Таблица 6.5 — Календарный ленточный график работ

| Стадия разработки   | Должность исполнителя | Трудоемкость |
|---------------------|-----------------------|--------------|
| Техническое задание | Ведущий программист   | 20           |
|                     | Программист 1         | 8            |
|                     | Программист 2         | 8            |
| Эскизный проект     | Ведущий программист   | 51           |
|                     | Программист 1         | 25           |
|                     | Программист 2         | 25           |
| Технический проект  | Ведущий программист   | 23           |
|                     | Программист 1         | 10           |
|                     | Программист 2         | 10           |
| Внедрение           | Ведущий программист   | 13           |
|                     | Программист 1         | 6            |
|                     | Программист 2         | 6            |

Исходя из табл. 6.5, при использовании оптимизации в результате параллельной работы ведущего инженера и программистов можно добиться сокращения срока разработки и внедрения программного продукта с 233 дней до 107 дней, т. е. в 2,17 раза.

## 6.2 Определение цены программной продукции

Затраты на выполнение проекта состоят из затрат на заработную плату исполнителям, затрат на закупку или аренду оборудования, затрат на организацию рабочих мест, и затрат на накладные расходы. Ниже приведены затраты на заработную плату и отчисления на социальное страхование в пенсионный фонд, фонд занятости и фонд обязательного медицинского страхования (26%). Для всех исполнителей предполагается оклад в размере 10000 рублей в месяц.

Таблица 6.6 — Затраты на зарплату и отчисления на социальное страхование

| Месяц   |              | Исполнитель |       |       | Итого, руб. |
|---------|--------------|-------------|-------|-------|-------------|
|         |              | 1           | 2     | 3     |             |
| Февраль | Рабочих дней | 20          | 8     | 8     | 18900       |
|         | Зарплата     | 8333        | 3333  | 3333  |             |
|         | ЕСН          | 2167        | 867   | 867   |             |
| Март    | Рабочих дней | 24          | 24    | 24    | 37800       |
|         | Зарплата     | 10000       | 10000 | 10000 |             |
|         | ЕСН          | 2600        | 2600  | 2600  |             |
| Апрель  | Рабочих дней | 24          | 1     | 1     | 13650       |
|         | Зарплата     | 10000       | 417   | 417   |             |
|         | ЕСН          | 2600        | 108   | 108   |             |
| Май     | Рабочих дней | 22          | 10    | 10    | 22050       |
|         | Зарплата     | 9167        | 4167  | 4167  |             |
|         | ЕСН          | 2383        | 1083  | 1083  |             |
| Май     | Рабочих дней | 17          | 6     | 6     | 15225       |
|         | Зарплата     | 7083        | 2500  | 2500  |             |
|         | ЕСН          | 2500        | 650   | 650   |             |
| Итого   |              |             |       |       | 107625      |

Расходы на материалы, необходимые для разработки программной продукции, указаны в табл. 6.7.

Таблица 6.7 — Затраты на материалы

| №     | Наименование                            | Ед. измерения | Количество | Цена за единицу, руб. | Сумма, руб. |
|-------|---|---------------|------------|-----------------------|-------------|
| 1     | Flash USB Drive                         | Шт.           | 1          | 400                   | 400         |
| 2     | Бумага А4                               | Пачка 500 л.  | 1          | 100                   | 100         |
| 3     | Картридж для принтера Samsung "ML-1210" | Шт.           | 1          | 2300                  | 2300        |
| Всего |   |               |            |                       | 2800        |

В работе над проектом используется специальное оборудование – персональные электронно-вычислительные машины (ПЭВМ) в количестве 3 шт. Стои-

мость одной ПЭВМ составляет 25000 рублей. Период амортизации 36 мес. Месячная норма амортизации рассчитывается по формуле:

$$K = \frac{1}{n} \cdot 100\% \quad (6.15)$$

Таким образом:

$$K = \frac{1}{36} \cdot 100\% = 2.78 \%$$

За 5 месяцев работы расходы на амортизацию составят:  $25000 \cdot 0.0278 \cdot 5 \cdot 3 = 10417$  руб.

Общие затраты на разработку ПП составят:  $107625 + 10417 = 118042$  рублей.

### 6.2.1 Расчет экономической эффективности

Расчет экономической эффективности проведем, предполагая цену разработанной программной продукции 320000 рублей. Основными показателями экономической эффективности является чистый дисконтированный доход (ЧДД) и срок окупаемости вложенных средств. Чистый дисконтированный доход определяется по формуле:

$$\text{ЧДД} = \sum_{t=0}^T (R_t - Z_t) \cdot \frac{1}{(1+E)^t} \quad (6.16)$$

где  $T$  – горизонт расчета по месяцам;

$t$  – период расчета;

$R_t$  – результат, достигнутый на  $t$  шаге (стоимость);

$Z_t$  – затраты;

$E$  – приемлемая для инвестора норма прибыли на вложенный капитал.

Коэффициент  $E$  установим равным ставке рефинансирования ЦБ РФ – 8% годовых (или 0.67% в месяц). В результате анализа рынка программной продукции с направленностью, аналогичной разрабатываемой, планируется продажа 2 единицы ПП каждый месяц. Планируемая цена ПП составляет 20000 рублей. Предполагаемые накладные расходы, связанные с реализацией составят 1500 рублей в месяц.

Коэффициент дисконтирования равен  $K_D = \frac{1}{(1+E)} = 0.9259$ .

В табл. 6.8 приведен расчет ЧДД по месяцам работы над проектом.



Таблица 6.8 — Расчет ЧДД

| Месяц    | Текущие затраты, руб. | Суммарные затраты, руб. | Текущий доход, руб. | ЧДД, руб.  |
|----------|-----------------------|-------------------------|---------------------|------------|
| Февраль  | 25172.22              | 25172.22                | 0                   | −25172.22  |
| Март     | 41272.22              | 66444.44                | 0                   | −63387.24  |
| Апрель   | 17122.22              | 83566.67                | 0                   | −78066.79  |
| Май      | 25522.22              | 109088.89               | 0                   | −98327.15  |
| Июнь     | 18697.22              | 127786.11               | 0                   | −112070.17 |
| Июль     | 1500                  | 129286.11               | 40000               | −85867.71  |
| Август   | 1500                  | 130786.11               | 40000               | −61606.18  |
| Сентябрь | 1500                  | 132286.11               | 40000               | −39141.8   |
| Октябрь  | 1500                  | 133786.11               | 40000               | −18341.45  |
| Ноябрь   | 1500                  | 135286.11               | 40000               | 918.13     |

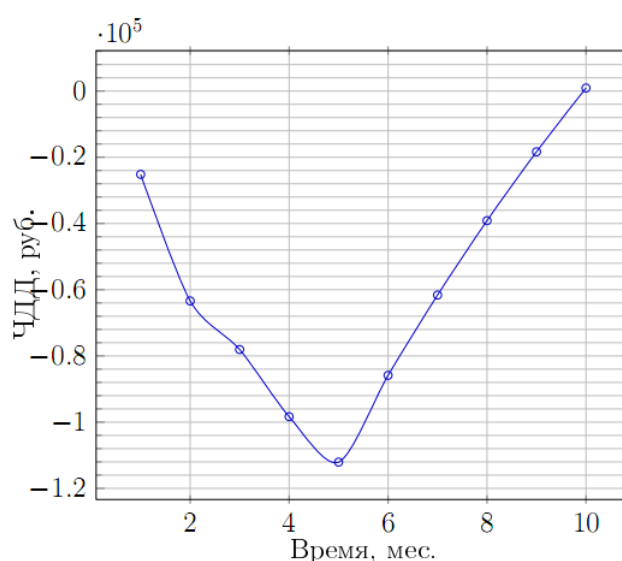


Рисунок 6.1 — График изменения чистого дисконтированного дохода

### 6.3 Вывод

Можно прогнозировать, что проект окажется рентабельным и окупится через 5 месяцев после внедрения. Созданный программный комплекс не имеет аналогов, реализующих все функциональные возможности комплекса. Существуют программные продукты, позволяющие производить голосовую аутентификацию, однако они нацелены на другой потребительский сектор. В силу вышесказанного, представляется довольно сложным оценить цену реализации программного средства при выдвигании на рынок. Автоматизация предметной области программного продукта началась относительно давно, рынок сбыта достаточно велик, и, предположения об окупаемости затрат на создание программы являются оптимистичными.

## Заключение

В результате выполнения проекта были выполнены следующие задачи:

- а) Спроектирован и реализован программный комплекс, предназначенный для голосовой аутентификации пользователя;
- б) Спроектирована и реализована подсистема сбора и хранения голосовых данных пользователя;
- в) Разработан алгоритм удаления из аудиозаписи фрагментов, не содержащих речь.
- г) Результаты проведенных экспериментальных исследований подтвердили практическую применимость разработанного программного обеспечения для защиты интернет-ресурса;

В качестве дальнейших путей развития системы можно рассматривать следующие:

- а) использование алгоритмов компенсации внешнего шума;
- б) использование комбинированных подходов к моделированию источника речи (нейросети, системы на основе нечетких множеств);
- в) создание специализированных клиентских приложений для предоставления возможности использования системы голосовой аутентификации пользователям мобильных устройств (коммуникаторы, карманные ПК).

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Bezdek J.C., Harris J.D.* Fuzzy portions and relations; an axiomatic basis for clustering // *Fuzzy Sets and Systems*. — 1978. — Vol. 1. — Pp. 111–127.
2. Biometric Systems: Worldwide Deployments, Market Drivers, and Major Players. — 2009.
3. The DET Curve in Assessment of Detection Task Performance / Alvin Martin, George Doddington, Terri Kamm et al. // *Proc. Eurospeech '97*. — Rhodes, Greece: 1997. — Pp. 1895–1898.
4. *Dempster A., Laird N., Rubin D.* Maximum likelihood from incomplete data via the EM algorithm // *J. Royal Stat. Society*. — 1977. — Vol. 39. — Pp. 1–38.
5. Discriminative training of GMM for speaker identification / C. M. del Alamo, F. J. Caminero Gil, C. dela Torre Munilla, L. Hernandez Gomez // *ICASSP '96: Proceedings of the Acoustics, Speech, and Signal Processing, 1996. on Conference Proceedings., 1996 IEEE International Conference*. — Washington, DC, USA: IEEE Computer Society, 1996. — Pp. 89–92.
6. *Falavigna Daniele*. Comparison Of Different HMM Based Methods For Speaker Verification.
7. *Fukunaga K.* Introduction to Statistical Pattern Recognition. — second edition edition. — London: Academic Press, 1990.
8. *Furui Sasaoki*. Cepstral analysis technique for automatic speaker verification // *IEEE Transactions on Acoustics, Speech, and Signal Processing*. — 1981. — Apr. — Vol. 29, no. 2. — Pp. 254–272.
9. *Greenwood Mark, Kinghorn Andrew*. SUVing: automatic Silence/Unvoiced/Voiced classification of speech. — 2000.
10. *Jayanna H.S., Prasanna S.R. Mahadeva*. Analysis, Feature Extraction, Modeling and Testing Techniques for Speaker Recognition // *IETE Tech Rev*. — 2009. — May. — Vol. 26, no. 3. — Pp. 181–190.
11. *Kinnunen Tomi, Karpov Evgeny, Franti Pasi*. Efficient Online Cohort Selection Method for Speaker Verification. — 2004.
12. *Lemire Daniel*. Faster Retrieval with a Two-Pass Dynamic-Time-Warping Lower Bound // *CoRR*. — 2008. — Vol. abs/0811.3301.

13. *Olsson Johan*. Text Dependent Speaker Verification with a Hybrid HMM/ANN System. — 2002.
14. *Pukelsheim Friedrich*. The Three Sigma Rule // *The American Statistician*. — 1994. — Vol. 48.
15. *Reynolds Douglas A*. Speaker identification and verification using Gaussian mixture speaker models // *Speech Commun.* — 1995. — Vol. 17, no. 1-2. — Pp. 91–108.
16. *Reynolds Douglas A., Quatieri Thomas F., Dunn Robert B*. Speaker verification using Adapted Gaussian mixture models // *Digital Signal Processing*. — 2000. — Pp. 19–41.
17. *Sakoe H., Chiba S*. Dynamic programming algorithm optimization for spoken word recognition // *IEEE Transactions on Acoustics, Speech, and Signal Processing*. — 1978. — Vol. 26, no. 1. — Pp. 43–49.
18. *Bachu R.G., Kopparthi S., Adapa B., Barkana B.D*. Separation of Voiced and Unvoiced using Zero crossing rate and Energy of the Speech Signal. — 2004.
19. *Tierney J*. A study of LPC analysis of speech in additive noise // *IEEE Transactions on Acoustics, Speech and Signal Processing*. — 1980. — Aug. — Vol. ASSP-28. — Pp. 389–397.
20. A Vector quantization approach to speaker recognition / F.K. Soong, A.E. Rosenberg, L.R. Rabiner, B.H. Juang // *IEEE Int. Conf. Acoust., Speech, Signal Process.* — Vol. 10. — Detroit, Michingan: 1985. — Apr. — Pp. 387–390.
21. Математическая статистика: Учеб. для вузов / В. Б. Горяинов, И. В. Павлов, Г. М. Цветкова и др.; Под ред. В. С. Зарубина, А. П. Крищенко. — 3-е изд. — М.: Изд-во МГТУ им. Н.Э. Баумана, 2008. — 424 с.
22. *Медведев М. С*. Использование вейвлет-преобразования для построения моделей фонем русского языка // *Вестник Сибирского государственного аэрокосмического университета имени академика М. Ф. Решетнева*. — 2006.
23. Теория вероятностей: Учеб. для вузов / А. В. Печинкин, О. И. Тескин, Г. М. Цветкова и др.; Под ред. В. С. Зарубина, А. П. Крищенко. — 3-е изд. — М.: Изд-во МГТУ им. Н.Э. Баумана, 2004. — 456 с.