

УДК 376

Илья Александрович Алексеев
Алина Александровна Вебер
Александр Алексеевич Копорулин
Юлия Сергеевна Пяшкур
г. Шадринск

Применение метода проектов при разработке приложения для альтернативного способа восприятия речи лицами с нарушениями слуха

В статье представлены этапы разработки приложения «Виброслух», в условиях реализации грантовой деятельности, на базе ФГБОУ ВО «Шадринский государственный педагогический университет» в научной лаборатории «Технологии диагностики и коррекции психоречевого развития ребенка». Программа «Виброслух», которая является частью программно-аппаратного комплекса, позволяет лицам с нарушениями слуха воспринимать устную речь через альтернативные каналы восприятия, такие как азбука Морзе, передаваемая вибрациями через специальные наушники с костной проводимостью.

Ключевые слова: лица с нарушениями слуха, азбука Морзе, метод проекта, программно-аппаратный комплекс, способы восприятия звучащей речи, язык программирования.

Ilya Alexandrovich Alekseev
Alina Alexandrovna Veber
Alexander Alekseevich Koporulin
Yulia Sergeevna Pyashkur
Shadrinsk

Application of the project method in the development of an application for an alternative way of speech perception by hearing impaired persons

The article presents the stages of the application development "Vibrohearing". The research has done as the implementation of grant activities, on the basis of Shadrinsk State Pedagogical University in the scientific laboratory "Technologies of diagnosis and correction of psychorechological development of a child". The "Vibrohearing" program, which is the part of the hardware and software complex, allows people with hearing impairments to perceive oral speech through alternative channels of perception such as Morse code transmitted by vibrations through special bone conduction headphones.

Keywords: hearing impaired persons, Morse code, project method, hardware and software complex, ways of perceiving sounding speech, programming language.

На современном этапе развития технологий и общества особое внимание уделяется созданию инновационных решений, направленных на улучшение качества жизни людей с ограниченными возможностями здоровья (ОВЗ). Важной задачей становится разработка приложений и устройств, которые позволяют интегрировать таких людей в социум, предоставляя им возможности для полноценного общения и восприятия окружающего мира. Одной из актуальных проблем, требующих решения, является обеспечение доступа к устной речи для лиц с тяжелыми нарушениями слуха. Решение данной проблемы возможно с применением метода проектов, который позволяет на практике обучать разработке программного обеспечения, создавая продукт, востребованный на современном рынке [2; 4].

В рамках деятельности научной лаборатории «Технологии диагностики и коррекции психоречевого развития ребенка» при ФГБОУ ВО «Шадринский государственный педагогический университет» и в рамках реализации грантовой деятельности, был осуществлен проект по организации междисциплинарного взаимодействия студентов, обучающихся по профилям: «Логопедия», «Дефектология» и «Программирование вычислительной техники и автоматизированных систем», а так же преподавателей вуза, специалистов, работающих в области дефектологии, информационных технологий и дизайна для решения научно-исследовательских и прикладных задач, направленных на разработку специализированного программно-аппаратного комплекса для восприятия устной речи

лицами с тяжелыми нарушениями слуха [1].

При реализации проекта, участниками группы было разработано приложение «Виброслух» – это система альтернативной коммуникации для распознавания голоса и перевода его в азбуку Морзе передаваемую на низких частотах (вибрациях) глухому человеку.

Для работы с программой пользователю необходимо иметь микрокомпьютер на базе операционной системы linux или windows, например raspberry, наушники способные воспроизводить низкие частоты (вибрации) и микрофон. Для функционирования приложения интернет не нужен, так как вся база данных хранится локально, для возможности использовать приложение в местах, где нет доступа в интернет.

Принцип работы приложения прост, пользователь включает Bluetooth наушники, включает микрокомпьютер, ждет сопряжения устройств между собой и может пользоваться. Далее приложение слышит голос и отбивает распознанный голос пользователю в низких частотах в виде азбуки Морзе. Исходя из выше сказанного, пользователь должен уметь распознавать азбуку Морзе.

Тестовый образец развернут на базе микрокомпьютера raspberry 4 с 4 гигабайтами оперативной памяти. Большой объем оперативной памяти обусловлен тем, что нейросеть, для распознавания голоса, выполняют все вычисления на устройстве, чем сильно расходует мощностные ресурсы. Для разработки использовался язык программирования python в связке с субдSQLITE и нейросетьюvosk. Данный язык программирования является более распространенным и подходящим для работы с нейросетями и устройствами на различных операционных системах, так как является интерпретируемым, что позволяет использовать его на любом устройстве, где есть интерпретатор, без перекомпиляции программного кода [3].

На данном этапе разработки приложение не имеет пользовательского интерфейса, но это не мешает его использовать по назначению.

Приложение «Виброслух» состоит из 5 частей:

1. Скрипт распознавания голоса.
2. Скрипт воспроизведения азбуки Морзе.
3. База данных SQLITE.
4. Звуковые файлы.
5. Скрипт автозапуска (Различных для операционных систем различного семейства).
6. Нейросеть VOSK.

Скрипт распознавания голоса работает следующим образом: перед началом работы происходит подключение всех необходимых библиотек для языка python. В эти библиотеки входят: библиотеки для работы с операционной системой, для работы с форматом json, для работы со звуком, для работы с реальным временем, библиотека нейросетивosk и базы данных SQLITE3 (рис. 1).

```
import os
import sys
import json
import pyaudio
import vosk
import time
import sqlite3
```

Рис. 1. Библиотека для языка python

Далее происходит подключение к базе данных, ее очистка с прошлой сессии и создание таблицы с необходимой структурой. Очистка с прошлой сессии нужна для того чтобы, распознанная речь, до текущего момента, не начала воспроизводиться при текущем запуске (рис.2).

```

conn = sqlite3.connect("speech.db")
cursor = conn.cursor()
cursor.execute("drop table recognized_text")
cursor.execute("CREATE TABLE IF NOT EXISTS recognized_text")
conn.commit()

```

Рис. 2. Работа с базой данных

Следующий шаг это генерация алфавита Морзе, для записи в базу данных (рис.3).

```

morze_alphabet = {
    "а": ".-", "б": "-...", "в": ".--", "г": "---", "д": "-..", "е": ".",
    "ж": "...-", "з": "--..", "и": "..", "й": ".---", "к": "-.-", "л": "-...-",
    "м": "--", "н": "-.", "о": "---", "п": "-.-.", "р": "-.-", "с": "...",
    "т": "-", "у": ".-.-", "ф": "-..-", "х": "....", "ц": "-.-.", "ч": "-.-.-",
    "ш": "----", "щ": "-.-.-", "ъ": "-.-.-", "ы": "-.-.-", "ь": "-.-.-",
    "э": "...-", "ю": "-.-.-", "я": "-.-.-"
}

```

Рис. 3. Генерация алфавита Морзе

Функция `text_to_morze` получает на вход распознанный текст, в соответствии с составленным алфавитом переводит его в текст азбуки Морзе. Перед переводом текста, функция сначала приводит весь полученный текст к нижнему регистру, для избегания проблем с работой алфавита (рис.4).

```

def text_to_morze(text):
    text = text.lower() # Приводим весь текст к нижнему регистру
    morse_result = []

    for char in text:
        if char in morze_alphabet:
            morse_result.append(morze_alphabet[char])
        elif char == " ":
            morse_result.append(" ") # Добавляем пробел для разделения слов
        else:
            morse_result.append("?") # Вопросительный знак для неизвестных символов

    return " ".join(morse_result) # Объединяем символы азбуки Морзе в строку

# Функция для записи текста в базу данных SQLite
def save_to_database(text, morze):
    try:
        timestamp = time.strftime('%Y-%m-%d %H:%M:%S')
        cursor.execute("INSERT INTO recognized_text (text, morze,timestamp,speech) VALUES (?,?,,?)",
            (text, morze, timestamp,0))
        conn.commit()
    except sqlite3.Error as e:
        print("SQLite error:", e)

```

Рис. 4. Функция `save_to_database` записывает в базу данных текст, строку азбуки Морзе и время когда произошла запись

Функция `main` является основной функцией, в которой происходит инициализация голосовой модели на русском языке, инициализация и настройка микрофона для получения голоса и запуск бесконечного цикла для распознавания. В бесконечном цикле происходит считывания аудиопотока с микрофона в реальном времени, распознавание текста с помощью нейросети и выполнение функций `text_to_morze` и `save_to_database` описанных выше. Скрипт воспроизведения азбуки Морзе предназначен для считывания данных с базы данных SQLITE3, и воспроизведения точек и тире в нужной последовательности обозначающих буквы азбуки Морзе.

В скрипте подключается схожий набор библиотек, в который еще входит библиотека для воспроизведения звуковых файлов (рис.5).

```
import os
import sys
import json
import pyaudio
import time
import pygame
import sqlite3
```

Рис. 5. Библиотека для воспроизведения звуковых файлов

Далее происходит подключение к базе данных, в которой хранится распознанный текст и строка азбуки Морзе для воспроизведения. Происходит инициализация библиотеки для воспроизведения звуков и ограничение каналов вывода звука. Подключаются сформированные аудиофайлы для точки и тире, и устанавливается значение громкости. Так как громкость на большинстве наушников можно регулировать непосредственно на них самих, поэтому значение громкости на максимум. Далее происходит генерация словаря для соответствий считывания точек и тире с базы данных (рис.6).

```
conn = sqlite3.connect("speech.db")
cursor = conn.cursor()

# Инициализация библиотеки pygame
pygame.mixer.init()
pygame.mixer.set_num_channels(1) # Ограничиваем количество каналов

# Загрузка аудиосэмплов для точки и тире
dot_sound = pygame.mixer.Sound("tochka_180.wav")
dash_sound = pygame.mixer.Sound("tire_160.wav")

# Регулирование громкости (от 0.0 до 1.0)
volume = 1
dot_sound.set_volume(volume)
dash_sound.set_volume(volume)

# Словарь для соответствия символов Морзе аудиосэмплам
morse_code_samples = {
    ".": dot_sound,
    "-": dash_sound
}
```

Рис. 6. Инициализация библиотеки для воспроизведения звуков и ограничение каналов вывода звука

Функция `play_morze_sound` – воспроизводит необходимый звук для соответствующего символа точки или тире. Функция `play_morze_code` – считывает текст из точек, тире и пробелов для воспроизведения каждого в нужной последовательности с помощью функции `play_morze_sound`, и регулирует необходимую задержку между символами и словами (рис.7).

```

def play_morse_sound(symbol):
    if symbol in morse_code_samples:
        sound = morse_code_samples[symbol]
        channel = sound.play()
        while channel.get_busy():
            pass

def play_morse_code(morse_string):
    for symbol in morse_string:
        if(symbol != " "):
            play_morse_sound(symbol)
            time.sleep(0.05)
        else:
            time.sleep(0.15)

```

Рис. 7. Функции play_morze_sound и play_morze_code

Основная функция main запускает бесконечный цикл, который считывает еще не воспроизведенные строки с базы данных, вызывает выше описанные функции для их воспроизведения и изменяет запись в базы данных, для пометки, что запись была воспроизведена (рис.8).

```

def main():
    try:
        while True:
            cursor.execute("Select * FROM recognized_text Where speech = 0 order by timestamp LIMIT 1")
            last_record = cursor.fetchone()
            if last_record:
                id, text, morze, timestamp, speech = last_record
                print(f"ID: {id}")
                print(f"Text: {text}")
                print(f"Morse Text: {morze}")
                print(f"Timestamp: {timestamp}")
                print(f"Speech: {speech}")
                if(morze):
                    cursor.execute("UPDATE recognized_text set speech=1 Where recognized_text.id = ?",
                                   (id,))
                    conn.commit()
                    play_morse_code(morze)
            else:
                print("База данных пуста или нет записей.")
                time.sleep(1)
    except KeyboardInterrupt:
        print("* stopped voice")

```

Рис. 8. Функция main запускает бесконечный цикл

Как говорилось ранее, скрипт для автозапуска приложения для каждого семейства операционных систем свой, все что он делает – это подключает Bluetooth устройство к микрокомпьютеру и запускает комплекс скриптов приложения «Виброслух».

На тестовом устройстве использовалась операционная система UbuntuServer 22.04, данная ОС не имеет графической оболочки, что позволяет экономить ресурсы устройства и отдавать предпочтение для использования приложения «Виброслух». Однако использование системы семейства linux без графической оболочки, на начальном этапе подготовки устройства, создает ряд сложностей, которых бы не возникло с операционной системой с графической оболочкой.

Первая сложность возникает в отсутствии ряда пакетов для работы с Bluetooth устройством, воспроизводящим и принимающим звук, так как по умолчанию разработчики считают, что использование подобных устройств на таких операционных системах не уместно. Это пакеты: bluez, bluez-tools, и pulseaudio – без которых устройство все же подключится, но воспроизводить и принимать звук не будет (рис. 9).

Следующая сложность заключается в том, что подключение Bluetooth устройства может осуществляться только после входа в систему. По умолчанию на системах, без

графической оболочки, автовход в операционную систему отключен, его следует сделать самостоятельно. Для операционных систем семейства linux необходимо выполнить следующие команды:

1) Создать директорию для конфигурации автовхода

```
sudo mkdir -p /etc/systemd/system/getty@tty1.service.d/
```

2) Создать файл сервиса

```
sudo nano /etc/systemd/system/getty@tty1.service.d/autologin.conf
```

И заполнить его содержимым сервиса автовхода указанного пользователя

```
[Service]
```

```
ExecStart=
```

```
ExecStart=-/sbin/agetty --autologinИМЯ_ПОЛЬЗОВАТЕЛЯ_СИСТЕМЫ --noclear %I
```

\$TERM

3) Перезапустить демона службы сервисов

```
sudo systemctl daemon-reload
```

4) Перезагрузить всю систему

```
sudo reboot
```



Рис. 9. Принцип работы приложения «Виброслух»

Таким образом, разработка данного приложения является актуальным и значимым вкладом в развитие использования альтернативных способов коммуникации для людей с ограниченными возможностями. Метод проектов, использованный в рамках разработки приложения, позволяет не только создать полезный продукт, но и обучить разработке программного обеспечения на практике, что важно для подготовки специалистов в области информационных технологий.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Алексеев, И.А. Учебный проект приложения для организации альтернативных способов коммуникации лиц с ОВЗ / И.А. Алексеев, А.А. Вебер, А.А. Копорулин. – Текст : непосредственный // Проблемы современного педагогического образования. – 2022. – № 77-4. – URL: https://www.elibrary.ru/download/elibrary_50105322_43649423.pdf (дата обращения: 20.10.2023).
2. Игнатова, Е.В. Особенности организации образовательного процесса для инвалидов и лиц с ОВЗ / Е.В. Игнатова, Ю.В. Игнатова. – Текст : непосредственный // Наука и образование. – 2023. – № 1. – С. 15-21.
3. Васильев, А. Программирование на Python в примерах и задачах / А. Васильев. – Москва : Эксмо, 2021. – 616 с. – Текст : непосредственный.

4. Щербакова, Я.Л. Нарушения слуха и методы их коррекции / Я.Л. Щербакова, Ю.К. Янов, В.Е. Кузовков, С.М. Мегрелишвили. – Текст : непосредственный // Российская оториноларингология. – 2014. – № 6 (73). – С. 104-110.

СВЕДЕНИЯ ОБ АВТОРАХ:

И.А. Алексеев, заведующий кафедрой коррекционной педагогики и специальной психологии, кандидат педагогических наук, доцент, ФГБОУ ВО «Шадринский государственный педагогический университет», г. Шадринск, Россия, e-mail: filologshgpi@mail.ru.

А.А. Вебер, старший преподаватель кафедры коррекционной педагогики и специальной психологии, научный сотрудник, ФГБОУ ВО «Шадринский государственный педагогический университет», г. Шадринск, Россия, e-mail: lina.veber.95@mail.ru.

А.А. Копорулин, инженер-программист вычислительного центра, ФГБОУ ВО «Шадринский государственный педагогический университет», г. Шадринск, Россия, e-mail: sanyakop98@yandex.ru.

Ю.С. Пяшкур, старший преподаватель кафедры коррекционной педагогики и специальной психологии, заместитель директора института психологии и педагогики по учебной работе, ФГБОУ ВО «Шадринский государственный педагогический университет», г. Шадринск, Россия, e-mail: dolgix_y-1485@mail.ru.

INFORMATION ABOUT THE AUTHORS:

I.A. Alekseev, h. D. in Pedagogical Sciences, Associate Professor, Department Chair, Department of Correctional Pedagogy and Special Psychology, Shadrinsk State Pedagogical University, Shadrinsk, Russia, e-mail: filologshgpi@mail.ru.

A.A. Veber, Senior Lecturer, Department of Correctional Pedagogy and Special Psychology, research associate, Shadrinsk State Pedagogical University, Shadrinsk, Russia, e-mail: lina.veber.95@mail.ru.

A.A. Koporulin, Software Engineer of the Computing Center, Shadrinsk State Pedagogical University, Shadrinsk, Russia, e-mail: sanyakop98@yandex.ru.

Y.S. Pyashkur, Senior Lecturer, Department of Correctional Pedagogy and Special Psychology, Deputy Director of the Institute of Psychology and Pedagogy for Academic Work, Shadrinsk State Pedagogical University, Shadrinsk, Russia, e-mail: dolgix_y-1485@mail.ru.