ЦЕЛЬ РАБОТЫ

Следует реализовать приложение на C99/C++11, которое должно уметь:

1. Получать данные из socket.
2. Передавать данные в socket.
3. Идентифицировать отправителя/получателя.
4. Вести историю передаваемых данных.

ОСНОВНАЯ ЧАСТЬ

1.1 Теоретическая часть

unix-socket (англ. Unix domain socket, UDS) — конечная точка обмена данными, подобная Интернет-сокету, но не использующая сетевой протокол для взаимодействия (обмена данными). Используется в операционных системах, поддерживающих стандарт POSIX, для межпроцессного взаимодействия.

Доменные соединения Unix являются по сути байтовыми потоками, сильно напоминая сетевые соединения, но при этом все данные остаются внутри одного компьютера (то есть обмен данными происходит локально). UDS используют файловую систему как адресное пространство имен, то есть они представляются процессами как иноды в файловой системе. Это позволяет двум различным процессам открывать один и тот же сокет для взаимодействия между собой. Однако, конкретное взаимодействие, обмен данными, не использует файловую систему, а только буферы памяти ядра.

1.2 Практическая часть

Было создано приложение на C99/C++11, которое действует по сценарию:

1. Сценарий сервиса по подсчету символов в слове, представленный в приложении А

1. Создается unix-сокет, связывается с `./socket` и начинает ждать подключений.
2. При подключении пользователя сервис просит его авторизоваться.
3. После авторизации пользователя сервис выдает правила своей работы и ждет ввода.

2. Сценарий пользователя, представленный в приложении Б

1. Обращается к созданному сервисом unix-сокету, связанному с `./socket`.
2. Вводит свой login по просьбе сервиса.
3. Производит ввод слов.

Есть три вида ввода пользователя:

- Слово - подсчет символов в данном слове. При вводе несколько слов (последовательность символов через пробел), сервис подсчитает символы в каждом слове отдельно.
- `logout` - отключение текущего пользователя и ожидание нового подключения.
- `end` - завершение работы сервиса.

Ввод пользователя и ответ сервиса выводятся в консоль в виде:

```
Service -> user : сообщение

user -> Service : сообщение
```

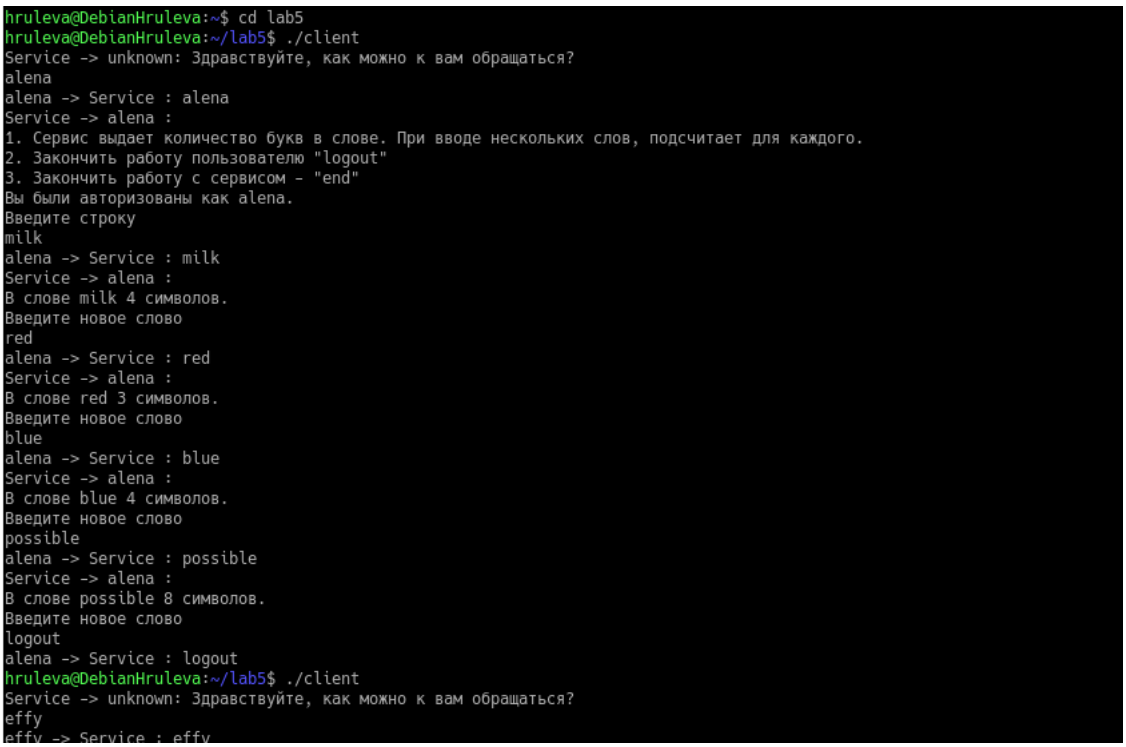

Рисунок 1 – Пример работы сервиса с пользователем

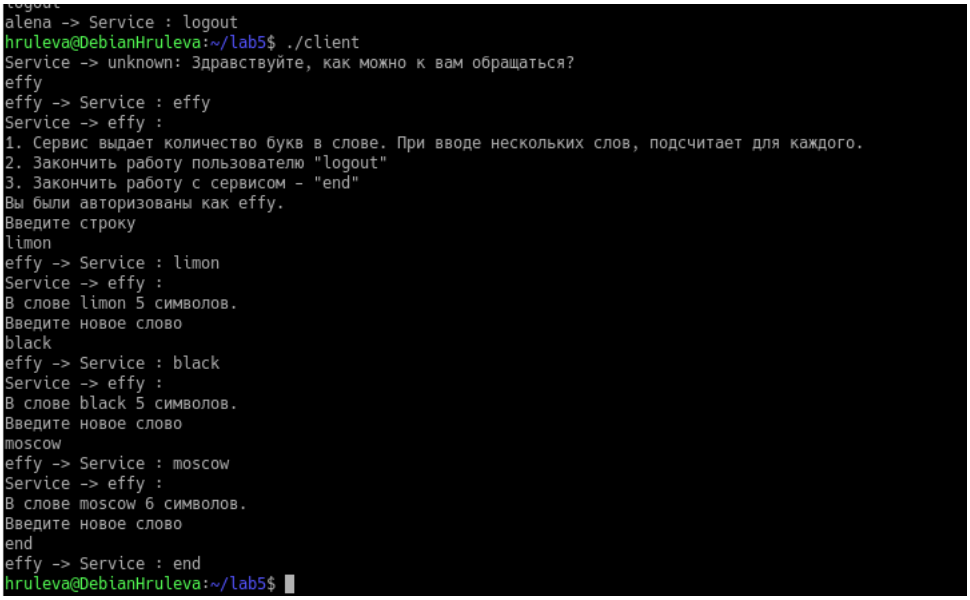

Рисунок 2 – Пример работы сервиса с пользователем

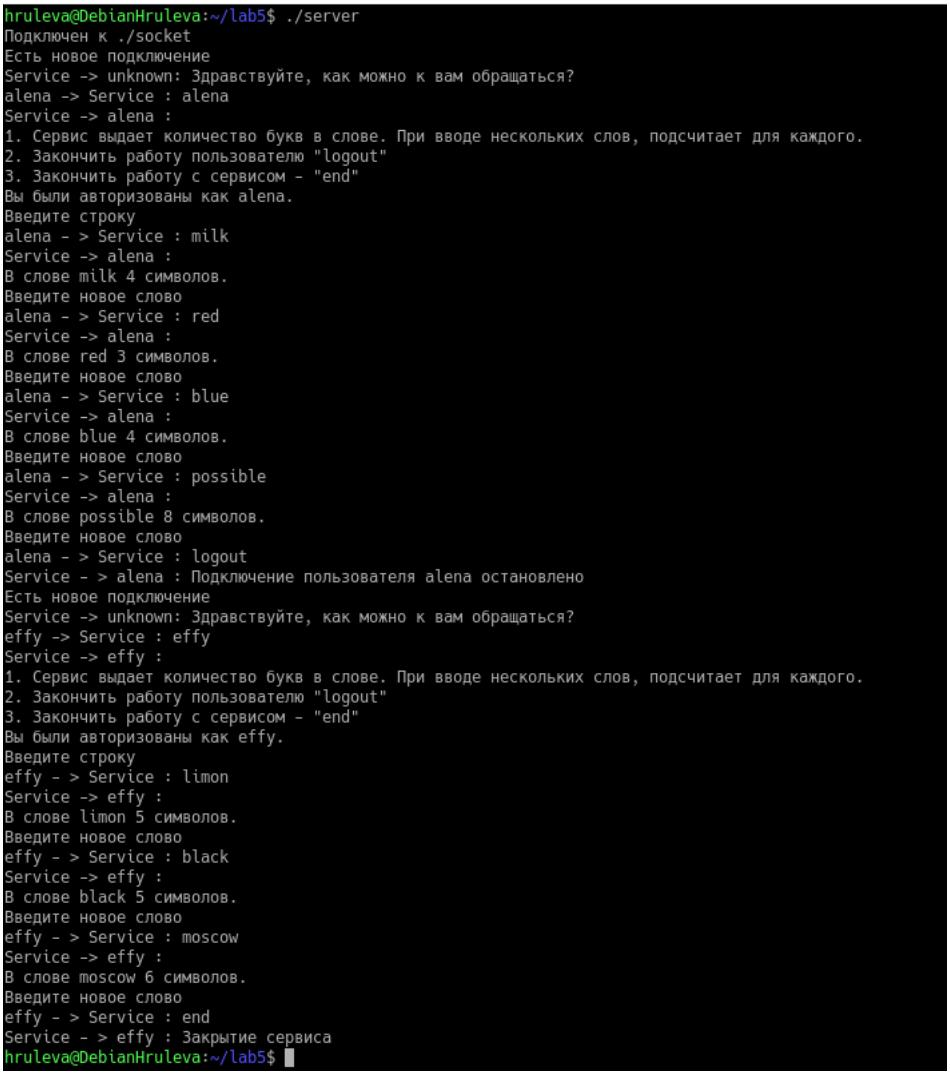

Рисунок 3 – Пример работы сервиса с пользователем

ВЫВОДЫ

Таким образом, в ходе лабораторной работы были изучены unix-sockets и реализовано приложение на C99/C++11 , осуществляющее работу с unix-socket.