Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №3

по «Низкоуровневому программированию»

Выполнил:

Студент группы P33302
Верзаков А.Ю.

Преподаватель:
Кореньков Ю.Д.

Санкт-Петербург

2023

Задание:

На базе данного транспортного формата описать схему протокола обмена информацией и воспользоваться существующей библиотекой по выбору для реализации модуля, обеспечивающего его функционирование. Протокол должен включать представление информации о командах создания, выборки, модификации и удаления данных в соответствии с данной формой, и результатах их выполнения.

Описание:

Программа состоит из модулей, полученных в результате лабораторной работы №1 (db) и №2 (analyzer). Для сериализации и передачи данных был реализован модуль pblib (на базе модуля protocol buffers).

Сборка осуществляется с помощью shell скрипта, который собирает файлы для анализатора (yacc и lex) и производит сборку исполняемых файлов посредством cmake.

В результате получаются 2 модуля serv и client – сервер и клиент соответственно.

Была использована библиотека nanopb (адаптация protocol buffers для C) и были реализованы следующие структуры для её работы (файл message.proto)

```
syntax = "proto2";

message Query {
    required int32 command = 1;
    repeated Filter filtersList = 2;
    repeated ValueSetting settingsList = 3;

    message Filter {
        repeated Comparator compList = 1;
    }

    message ValueSetting {
        required KeyValuePair kv = 1;
    }

    message Comparator {
        required int32 operation = 1;
        required KeyValuePair kv = 2;
    }

    message KeyValuePair {
        required string key = 1;
        required int32 valueType = 2;
        required int64 valueInt = 3;
        required float valueReal = 4;
        required string valueString = 5;
    }
}

message Response {
    required int32 last = 2;
    required string rString = 1;
}
```

Результат запроса (ответ от сервера) – это строка с результатами выполнения, либо описанием ошибки. Ответ разбивается на пакеты по 1024 байта, которые отправляются по очереди, если ответ оказывается большим.

Передача организована стандартным образом, с помощью OS API (sockets)

Пример со стороны клиента:

```
int main (int c, char** v) {

    if (c != 2) {
        printf("Only 1 argument expected: host address\n");
    }

    struct sockaddr_in local;
    char* path = NULL;

    if (c > 1) path = v[1];

    socket = socket(AF_INET, SOCK_STREAM, 0);

    memset(&local, 0, sizeof(local));
    local.sin_family = AF_INET;
    local.sin_addr.s_addr = inet_addr(v[1]);
    local.sin_port = htons(PORT);

    if (connect(socket, (struct sockaddr*) &local, sizeof(local)) != 0) {
        perror("Connection error!");
        return 1;
    }

    printf("Client: connecting...\n");

    while (1) {
        yyparse();
    }

    close(socket);

    return 0;
}
```

…

Со стороны сервера:

```
listenFd = socket(AF_INET, SOCK_STREAM, 0);
setsockopt(listenFd, SOL_SOCKET, SO_REUSEADDR, &reuse, sizeof(reuse));

memset(&serverAddress, 0, sizeof(serverAddress));
serverAddress.sin_family = AF_INET;
serverAddress.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
serverAddress.sin_port = htons(PORT);

if (bind(listenFd, (struct sockaddr*)&serverAddress, sizeof(serverAddress)) != 0) {
    perror("binding error");
    return 1;
}

if (listen(listenFd, 5) != 0) {
    perror("listening error");
    return 1;
}
```

…

Примеры выполнения запросов:

```
storage.receive({pName: "BIOS"})
--- FIND RESULT ---
--- TUPLE  49 ---
pName                BIOS
name                 SystemVersion
value                V1.29
--- TUPLE  48 ---
pName                BIOS
name                 SystemSKU
value                0
--- TUPLE  47 ---
pName                BIOS
name                 SystemProductName
value                Nitro_AN515-54
--- TUPLE  46 ---
pName                BIOS
name                 SystemManufacturer
value                Acer
--- TUPLE  45 ---
pName                BIOS
name                 SystemFamily
value                Nitro_5
--- TUPLE  44 ---
pName                BIOS
name                 BIOSVersion
value                V1.29
--- TUPLE  43 ---
pName                BIOS
name                 BIOSVendor
```

```
storage.insert({parent: 15}, {pName: "System", name: "Windows", value: "a10"})
Added successfully.
storage.receive({pName: "System"})
--- FIND RESULT ---
--- TUPLE  51 ---
pName                System
name                 Windows
value                a10
```

```
storage.update({name: "HARDWARE"}, $set: {value: "software"})
--- UPDATE RESULT ---
Updated id: 0
storage.receive({value: "software"})
--- FIND RESULT ---
--- TUPLE   0 ---
pName                HARDWARE
name                 HARDWARE
value                software
```

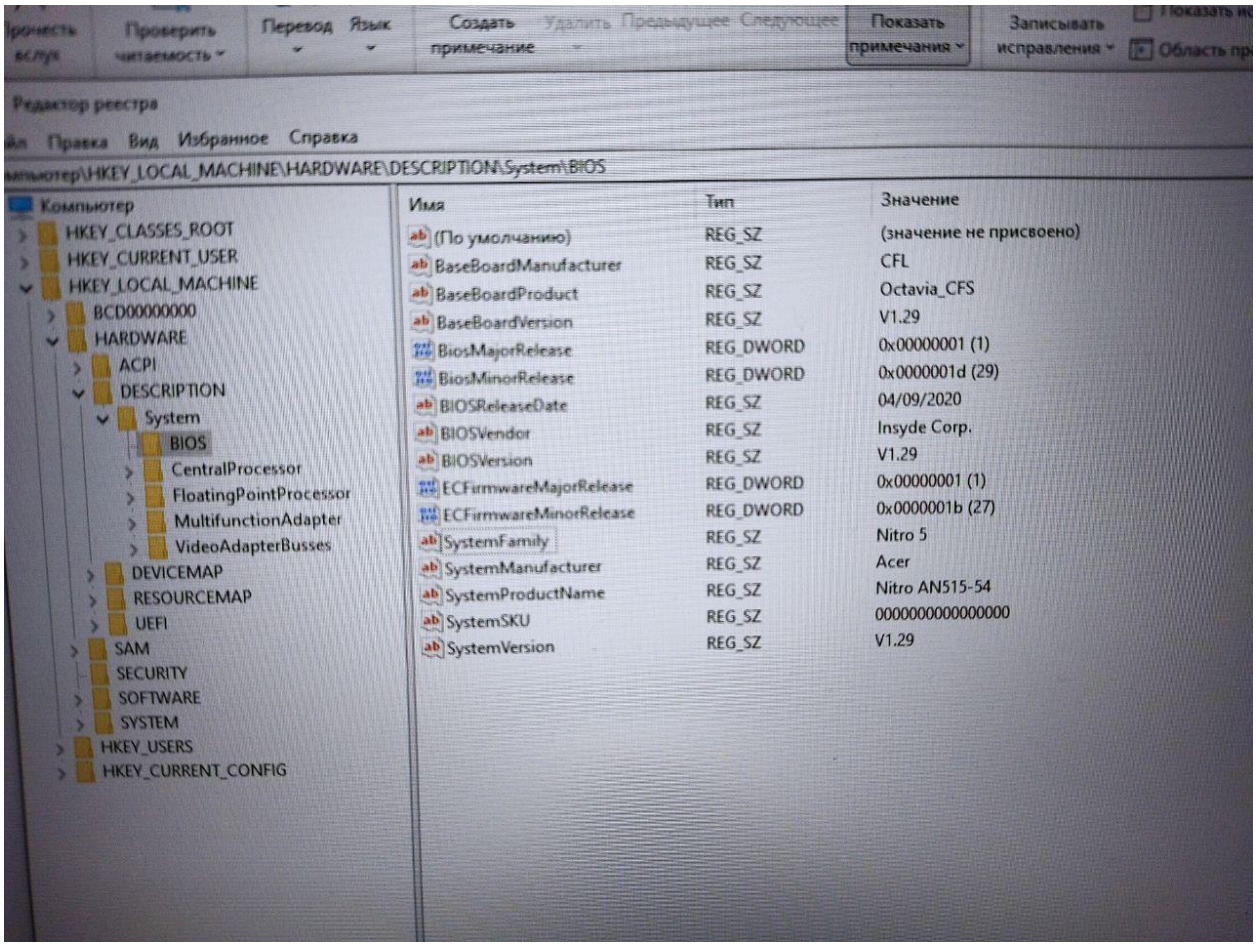
```
storage.delete({value: "software"})
--- REMOVE RESULT ---
Removed successfully id: 0
storage.receive({value: "software"})
--- NO RESULTS ---
```

Исходные данные брались из реестра Windows, было выполнено дополнительное задание – вывести поля вершины “BIOS” из реестра.

Результаты выполнения:

```
storage.receive({pName: "BIOS"})
--- FIND RESULT ---
--- TUPLE  49 ---
pName                BIOS
name                 SystemVersion
value                V1.29
--- TUPLE  48 ---
pName                BIOS
name                 SystemSKU
value                0
--- TUPLE  47 ---
pName                BIOS
name                 SystemProductName
value                Nitro_AN515-54
--- TUPLE  46 ---
pName                BIOS
name                 SystemManufacturer
value                Acer
--- TUPLE  45 ---
pName                BIOS
name                 SystemFamily
value                Nitro_5
--- TUPLE  44 ---
pName                BIOS
name                 BIOSVersion
value                V1.29
--- TUPLE  43 ---
pName                BIOS
name                 BIOSVendor
value                Insyde_Corp.
--- TUPLE  42 ---
pName                BIOS
name                 BIOSRealiseDate
value                04/09/2020
--- TUPLE  41 ---
pName                BIOS
name                 BaseBoardVersion
value                V1.29
--- TUPLE  40 ---
pName                BIOS
name                 BaseBoardProduct
value                Octavia_CFS
--- TUPLE  39 ---
pName                BIOS
name                 BaseBoardManufacturer
value                CFL
--- TUPLE  38 ---
pName                BIOS
name                 BIOS
```

Значения вершин из реестра:



Результаты совпадают

Вывод:

В ходе работы я интегрировал стороннюю библиотеку в свой проект, соединил воедино отдельные модули клиента и сервера. Была реализована передача данных (порой довольно большого объема) на низком уровне за оптимальное время. Также я научился програмно работать с реестром системы Windows, в ходе выполнения дополнительного задания.