Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление системного и прикладного программного обеспечения

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

курса «Низкоуровневое программирование»

по теме: «Клиент-серверное взаимодействие»

Вариант № 2

Выполнил студент:
Тюрин Иван Николаевич
группа: P33102

Преподаватель:
Кореньков Ю. Д.

Санкт-Петербург, 2024 г.

Содержание

Лабораторная работа № 3 Клиент-серверное взаимодействие

1. Задание варианта № 2

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.

Используя созданные в результате выполнения заданий модули, разработать в виде консольного приложения две программы: клиентскую и серверную части. Серверная часть – получающая по сети запросы и операции описанного формата и последовательно выполняющая их над файлом данных с помощью модуля из первого задания. Имя фала данных для работы получать с аргументами командной строки, создавать новый в случае его отсутствия. Клиентская часть – в цикле получающая на стандартный ввод текст команд, извлекающая из него информацию о запрашиваемой операции с помощью модуля из второго задания и пересылающая её на сервер с помощью модуля для обмена информацией, получающая ответ и выводящая его в человеко-понятном виде в стандартный вывод.

Формат транспортного протокола: `JSON`.

2. Выполнение задания

В процессе работы были выполнены следующие ключевые задачи:

1. Спроектирован формат запросов и ответов
2. Спроектированы JSON схемы для представления запросов и ответов
3. В результате длительного поиска и изучения информации. Выбрана библиотека C++ nlohmann/json как наиболее удобная для выполнения поставленной задачи.
4. Для кроссплатформенной работы разработана обертка над API сокетов в системах Windows и *Nix (TcpSocket).
5. Созданы реализации (при помощи препроцессора выбирается какой .cpp файл использовать) для Windows и *Nix
6. Созданы и реализованы классы Server и Client отвечающие соответственно за работу серверного и клиентского приложений.
7. Клиентское и серверное прилижения используют модули для хранения данных и синтаксического разбора запроса, разработанные в предыдущих лабораторных работах.

3. Описание работы

Проект состоит из следующих модулей, опуская модули из предыдущих лабораторных работ:

- `client` - содержит исходный код клиентского приложения, который делает синтаксический разбор запросов XPath-подобного синтаксиса в AST, переводит AST в JSON, валидирует полученный запрос, отправляет его серверу, ожидает результат в ответ в формате JSON. Полученный ответ, он десереализует в объект порции данных для дальнейшей работы с ним, например, вывода на экран.
- `server` - содержит исходный код серверного приложения, которое принимает поддерживает соединение, принимает запросы, которые валидирует разработанной JSON схемой запроса, и передает на дальнейшую обработку классу StorageAdapter, который выполняет соответствующие запросы с хранилищем и возвращает результат в виде JSON обратно. Полученный результат сервер отправляет обратно клиенту по сети в соответствии со схемой ответа.

- `common` - содержит общие для клиента и сервера заголовки структур данных и класс TcpSocket - обертку на Socket API, и платформно-зависимые реализации.

3. 1. JSON схема запроса

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "$id": "./query_schema.json",
    "title": "Query",
    "description": "Query json represantation",
    "definitions": {
        "element": {
            "type": "object",
            "properties": {
                "elementType": {
                    "type": "string"
                },
                "attributes": {
                    "type": "array",
                    "items": {
                        "oneOf": [
                            {
                                "type": "string"
                            },
                            {
                                "type": "number"
                            },
                            {
                                "type": "boolean"
                            }
                        ]
                    }
                }
            },
            "required": [
                "elementType",
                "attributes"
            ],
            "additionalProperties": false
        },
        "attributesPattern": {
            "oneOf": [
                {
                    "description": "binary operation",
                    "type": "object",
                    "properties": {
                        "patternType": {
                            "const": "BIN_OP"
                        },
                        "operation": {
                            "enum": [
                                "AND",
                                "OR"
                            ]
                        },
                        "lhs": {
                            "$ref": "#/definitions/attributesPattern"
```

```
                        },
                        "rhs": {
                            "$ref": "#/definitions/attributesPattern"
                        }
                    },
                    "required": [
                        "patternType",
                        "operation",
                        "lhs",
                        "rhs"
                    ],
                    "additionalItems": false
                },
                {
                    "description": "condition node",
                    "type": "object",
                    "properties": {
                        "patternType": {
                            "const": "COND"
                        },
                        "operation": {
                            "enum": [
                                "EQ",
                                "NEQ",
                                "GT",
                                "LT",
                                "GTE",
                                "LTE"
                            ]
                        },
                        "index": {
                            "type": "integer"
                        },
                        "value": {
                            "oneOf": [
                                {
                                    "type": "number"
                                },
                                {
                                    "type": "string"
                                }
                            ]
                        }
                    },
                    "required": [
                        "patternType",
                        "operation",
                        "index",
                        "value"
                    ],
                    "additionalItems": false
                },
                {
                    "description": "condition node",
                    "type": "object",
                    "properties": {
                        "patternType": {
                            "const": "COND"
                        },
                        "operation": {
```

```
                            "const": "CONTAINS"
                        },
                        "index": {
                            "type": "integer"
                        },
                        "value": {
                            "type": "string"
                        }
                    },
                    "required": [
                        "patternType",
                        "operation",
                        "index",
                        "value"
                    ],
                    "additionalItems": false
                }
            ]
        },
        "linkPattern": {
            "type": "object",
            "properties": {
                "linkType": {
                    "type": "string"
                },
                "target": {
                    "$ref": "#/definitions/elementPattern"
                }
            },
            "required": [
                "linkType"
            ],
            "additionalProperties": false
        },
        "elementPattern": {
            "type": "object",
            "properties": {
                "elementType": {
                    "type": "string"
                },
                "attributes": {
                    "$ref": "#/definitions/attributesPattern"
                },
                "linksIn": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/linkPattern"
                    }
                },
                "linksOut": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/linkPattern"
                    }
                }
            },
            "required": [
                "elementType",
                "linksIn",
                "linksOut"
```

```
            ],
            "additionalProperties": false
        }
    },
    "type": "object",
    "oneOf": [
        {
            "properties": {
                "queryType": {
                    "const": "create_vertex"
                },
                "element": {
                    "$ref": "#/definitions/element"
                }
            },
            "required": [
                "queryType",
                "element"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "enum": [
                        "drop_vertex",
                        "match"
                    ]
                },
                "pattern": {
                    "$ref": "#/definitions/elementPattern"
                }
            },
            "required": [
                "queryType",
                "pattern"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "create_edge"
                },
                "linkType": {
                    "type": "string"
                },
                "sourcePattern": {
                    "$ref": "#/definitions/elementPattern"
                },
                "dstPattern": {
                    "$ref": "#/definitions/elementPattern"
                }
            },
            "required": [
                "queryType",
                "linkType",
                "sourcePattern",
                "dstPattern"
            ],
```

```
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "update_vertex"
                },
                "pattern": {
                    "$ref": "#/definitions/elementPattern"
                },
                "attributes": {
                    "type": "array",
                    "items": {
                        "type": "object",
                        "properties": {
                            "index": {
                                "type": "integer"
                            },
                            "value": {
                                "oneOf": [
                                    {
                                        "type": "number"
                                    },
                                    {
                                        "type": "string"
                                    },
                                    {
                                        "type": "boolean"
                                    }
                                ]
                            }
                        },
                        "required": [
                            "index",
                            "value"
                        ],
                        "additionalProperties": false
                    }
                }
            },
            "required": [
                "queryType",
                "pattern",
                "attributes"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "create_vertex_type"
                },
                "attributes": {
                    "type": "array",
                    "items": {
                        "enum": [
                            "INT32",
                            "INT64",
                            "DOUBLE",
                            "BOOL",
```

```
                            "STRING"
                        ]
                    }
                },
                "elementType": {
                    "type": "string"
                }
            },
            "required": [
                "queryType",
                "attributes",
                "elementType"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "create_edge_type"
                },
                "linkType": {
                    "type": "string"
                }
            },
            "required": [
                "queryType",
                "linkType"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "drop_vertex_type"
                },
                "elementType": {
                    "type": "string"
                }
            },
            "required": [
                "queryType",
                "elementType"
            ],
            "additionalProperties": false
        },
        {
            "properties": {
                "queryType": {
                    "const": "drop_edge_type"
                },
                "linkType": {
                    "type": "string"
                }
            },
            "required": [
                "queryType",
                "linkType"
            ],
            "additionalProperties": false
        }
```

```
    ]
}
```

Листинг 1.1: Схема запроса

3. 2. JSON схема ответа

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "$id": "./response_schema.json",
    "title": "Response",
    "description": "Response json represantation",
    "type": "object",
    "properties": {
        "error": {
            "enum": [
                0,
                1
            ]
        },
        "result": {
            "type": "array",
            "items": {
                "type": "object",
                "properties": {
                    "elementType": {
                        "type": "string"
                    },
                    "attributes": {
                        "type": "array",
                        "items": {
                            "oneOf": [
                                {
                                    "type": "object",
                                    "properties": {
                                        "attributeType": {
                                            "enum": [
                                                "int64",
                                                "int32",
                                                "bool"
                                            ]
                                        },
                                        "value": {
                                            "type": "integer"
                                        }
                                    },
                                    "additionalProperties": false,
                                    "required": [
                                        "attributeType",
                                        "value"
                                    ]
                                },
                                {
                                    "type": "object",
                                    "properties": {
                                        "attributeType": {
                                            "const": "double"
                                        },
```

```
                                        "value": {
                                            "type": "number"
                                        }
                                    },
                                    "additionalProperties": false,
                                    "required": [
                                        "attributeType",
                                        "value"
                                    ]
                                },
                                {
                                    "type": "object",
                                    "properties": {
                                        "attributeType": {
                                            "const": "string"
                                        },
                                        "value": {
                                            "type": "string"
                                        }
                                    },
                                    "additionalProperties": false,
                                    "required": [
                                        "attributeType",
                                        "value"
                                    ]
                                }
                            ]
                        }
                    },
                    "required": [
                        "elementType",
                        "attributes"
                    ],
                    "additionalProperties": false
                }
            }
        },
        "required": [
            "error"
        ],
        "additionalProperties": false
}
```

Листинг 1.2: Схема ответа

4. Результат работы приложения

Приложение было протестировано, пример сеанса работы можно видеть на листинге:

```
>db.insert({pole/vtoroe_pole=true})
Ok
>db.find({pole/vtoroe_pole})
Ok: true
>db.insert({pole/vtoroe_pole=false})
Ok
```

```
>db.find({pole/vtoroe_pole})
Ok: false
>db.insert({pole/tretye_pole="lol"})
Ok
>db.find({pole/tretye_pole})
Ok: "lol"
>db.find({pole/*})
Ok: Count elements: 2
vtoroe_pole=false
tretye_pole="lol"
>db.delete({pole/vtoroe_pole=true})
Fail: No such node
>db.delete({pole/vtoroe_pole=false})
Ok
>db.find({pole/*})
Ok: Count elements: 1
```

Листинг 1.3: Пример сеанса работы

На листинге видно *безсхемную* натуру базы данных: каждое значение, которое записывается в нее, является узлом в дереве, на ровне с узлами-объектами (как в JSON).

5. Вывод

В результате выполнения работы удалось разработать базу данных, имеющую в основе документное дерево в качестве формы данных. При этом в работе использовались ранее разработанные модули для хранения данных и синтаксического разбора текстовых запросов. Приложение определенно нуждается в дальнейшем улучшении для его повсеместного использования. Во время выполнения работы были получены полезные навыки разработки сложной программной системы, укреплены навыки программирования на языке C++ и навыки работы с различными операционными системами, в том числе работа с вводом-выводом через отображение файла в виртуальное адресное пространство.

Исходный код проекта размещен в личном репозитории на сайте GitHub https://github.com/e1turin/itmo-llp/tree/lab-3.