

## 1. Система ранжирования

## 2. Зависимости

- C++14
- UbjsonCpp - библиотека для работы протокола (repo: <https://github.com/NightCodersss/>)
- cppunit - библиотека для unit-тестирования (используется в Ubjson)
- json - библиотека для чтения конфига (repo: <https://github.com/nlohmann/json>)

## 3. Разработчику

Помимо информации, изложенной здесь, следует изучить документацию, сгенерированную doxygen'ом (директории latex и html в корне проекта).

### 3.1. Структура

Структура выглядит так: (Фронт-энд или другой сервис)  $\xrightarrow{\text{Southernprotocol}}$  Ranking System  $\xrightarrow{\text{Northernprotocol}}$  IndexServer Где '  $\rightarrow$ ' обозначает подключение от клиента к серверу.

### 3.2. Протокол

В связи с тем, что Ubjson парсится на лету, не все поля обязательны. В силу гибкости протокола незначительное изменение или значительное добавление не должно требовать больших трудов. Что такое южный и северный легко понять отсюда ??.

#### 3.2.1. Южный

Запрос.

Необходимо передавать строчку query, которая хранит в себе запрос. Поле *amount* не обязательно.

```
{
"query": "what is ...?"
"amount": "1000000"
}
```

Ответ.

```
{  
  "docs": [ {...}, {...}, ],  
  "amount": 12243,  
}
```

Планируется добавить параметры, каким образом представлять информацию о документе обратно.

Заглушка работает так: слушает некоторый порт, по приходу данных, до переноса строки ( $\backslash n$ ), приходящие данные пакует в ubjson и отправляет системе ранжирования. По приходу данных от системы ранжирования (то есть системы поиска)(в ubjson) заглушка "их отвечает" в json (обычном).

### 3.2.2. Северный

Запрос.

Такой же, как "южный но с параметром - какой индекс использовать (по какому тексту), то есть:

```
{  
  "query": "what is ...?",  
  "index_id": "aa1234df",  
  "fields": ["docname", "author", "snippet"],  
}
```

Ответ.

Такой же, как у южного.

### 3.2.3. Реализация

Ясно, что нехорошо ждать каждого ответа. Глобально, есть два пути решения этой проблемы: использование асинхронных операций ввода-вывода (с callback'ами) и использование большого количества потоков. Существенных различий нет (нам известных), за исключением того, что код с большим количеством callback'ов плохо читаем и некрасив.

Итак, реализация такова, что на каждое соединение выделяется отдельный поток, в котором соединение работает синхронно.

## 3.3. Конфигурация

Следует обратить внимание, что конфигурация считывается в json, а не в ubjson. См. "конфигурация" для человека, который запускает.

## 4. Программистам-пользователям

См. структуру ??

## 5. Человеку, который запускает

См. структуру ??

### 5.1. Конфигурация

Очень важно соблюдать типы значений: строки должны быть в кавычках, числа без. Это не проверяется и приводит к повисанию (бага библиотеки работы с json). Пример конфига RankingServer'a:

```
{
  "texts": [
    {
      "servers": [{"host": "localhost", "port": "14000"}],
      "factor": {"person": 1.8, "article": 1.2},
      "name": "Title",
      "index_id": "1",
    },
    {
      "servers": [{"host": "", "port": "14000"}],
      "factor": 0.9,
      "name": "Useless information",
      "index_id": "100400ab",
    },
  ]
}
```