Схема организации клиент-серверного взаимодействия

**Приложение-клиент**

**(требуется JVM)**

Клиент 1



Клиент 2



Клиент 3

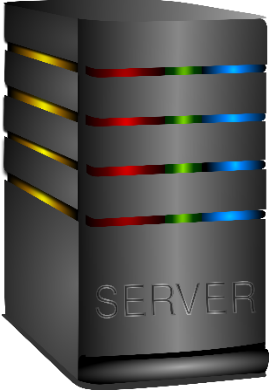


Клиент N



**Приложение-сервер**

**(требуется JVM)**



**Сетевое хранилище**



**БД (SQL) для хранения данных пользователя и прав доступа**



|  |  |
| --- | --- |
| **Вопрос** | **Ответ** |
| 1. Как организовать клиент-серверное взаимодействие? | 1. Создать сервер  2. Создать 2-3 клиента  3. Настроить взаимодействие между сервером и клиентами посредством протокола (json)  4. Сымитировать множественное обращение клиентов к серверу  5. Настройка авторизации, аутентификации и шифрования данных  Используется Java фреймворк Netty, nio, io |
| 2. Как и в каком виде передавать файлы? | Как – с использованием встроенных в Java методов?  В каком виде – as is, или вопрос про шифрование?   Оптимально: «Передавать сериализованные объекты, содержащие информацию о файлах»   «Как параллельно с файлами передавать служебные сообщения/команды? В отдельном сетевом потоке?» Какие есть best practice?   «Давать ли клиенту возможность передавать несколько файлов параллельно?» - давать. |
| 3. Как пересылать большие файлы? | Использование блочного метода передачи с:  - разбиением процесса передачи на несколько потоков  - с контрольной сумой для проверки целостности полученного файла |
| 4. Как пересылать служебные команды? | См. п.2 |
| 5. Что хранить в базе данных? | Хранить в БД данные пользователей (клиентов, users):  login, pwd hash, настройки доступа (admin, R, W, RW, no access) |
| 6. Как передавать структуру каталогов/файлов? | Создание дерева каталогов, исходя из потребностей пользователей и с соответствующими правами  Передача производится в формате .xml или json |
| 7. Какую библиотеку использовать для сетевого взаимодействия: java.io, java.nio, Netty? | См. п.1 |