Андрей Козлов, гр. 4538

**Централизованная система распространения и сдачи домашних заданий.**

1. **Постановка задачи**

Типичный цикл работы с домашними заданиями устроен следующим образом:

1. преподаватель задает домашнее задание по читаемому курсу;
2. студенты выполняют задание и сдают работы преподавателю;
3. преподаватель проверяет сданные работы и сообщает результат проверки студентам.

Реализованная система позволяет упросить процесс сдачи домашних заданий: преподаватели и студенты при помощи программы-клиента отправляют информацию о заданиях/решениях/результатах на сервер, где она сохраняется. При подключении нового клиента сервер пересылает ему всю необходимую информацию.

1. **Аутентификация пользователя**

При подключении к серверу клиент должен пройти аутентификацию.

Информация о преподавателях и студентах хранится в базе данных на сервере. Она включает в себя следующие текстовые поля:

1. имя и фамилия пользователя, для преподавателей название читаемого курса;
2. логин;
3. пароль.

Для аутентификации на сервере клиент отправляет логин и некоторый хеш пароля. Если аутентификация успешно пройдена, клиенту отправляется информация об имени и фамилии. Преподавателям также отправляется информация о названии читаемого курса.

1. **Модель**

Формализуем несколько понятий, которыми будем пользоваться в дальнейшем.

* *Задание* (**Task**) – сформулированное преподавателем домашнее задание.

Задание состоит из:

1. названия курса **courseName** (текст);
2. названия задания **taskName** (текст);
3. описания задания **text** (текст);
4. крайнего срока сдачи **deadline** (дата).

* *Решение* (**Solution**) – выполненная студентом работа по решению *задания*. Отправляется студентом преподавателю.

Решение состоит из:

1. имени студента **studentName** (текст);
2. названия курса **courseName** (текст);
3. названия задания **taskName** (текст);
4. выполненной работы **file** (файл).

* *Вердикт* (**Verdict**) – результат проверки преподавателем *решения*. Отправляется преподавателем студенту.

Вердикт состоит из:

1. имени студента **studentName** (текст);
2. названия задания **taskName** (текст);
3. флага, принято ли задание, **isAccepted** (логический тип);
4. комментариев **comments** (текст).

* *Запрос на аутентификацию* (**AuthenticationRequest**) – запрос клиента для входа на сервер.

Запрос состоит из:

1. логин пользователя **login** (текст);
2. хеш пароля **passwordHash** (число).

* *Ответ на запрос на аутентификацию* (**AuthenticationResponse**) – ответ сервера на *запрос*.

Ответ состоит из:

1. флаг, пройдена ли аутентификация, **isPassed** (логический тип);
2. имя клиента **name** (текст, информация о читаемом курсе включается в это поле).
3. **Репликация данных**

Для увеличения работоспособности системы поддерживается репликация данных. Система имеет два активных сервера, синхронизованных между собой. В случае выхода из строя одного из них работа системы не нарушается. Клиенты могут подключаться к любому из доступных серверов.

1. **Формирование сообщений**

Все сообщения в системе формируются по следующей схеме:

1. длина сообщения в байтах (4 байта);
2. тип сообщения (1 байт):
   * 0x01 – запрос на аутентификацию;
   * 0x02 – ответ на запрос на аутентификацию;
   * 0x03 – сообщение с решением;
   * 0x04 – сообщение с вердиктом;
   * 0x05 – сообщение с заданием;
3. поля объекта модели.
4. **Формирование массива байтов по полю объекта**

Значения примитивных типов записываются соответствующей последовательностью байт.

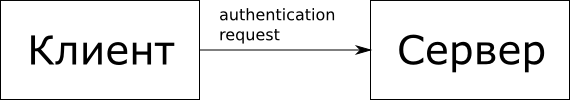
Массив байтов записывается следующим образом:

1. длина массива **length** (4 байта);
2. содержимое массива (length байт).

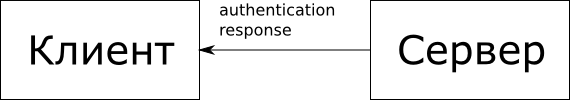
Текст записывается в виде массива байтов (кодировка строк UTF-8).

Файлы записываются следующим образом:

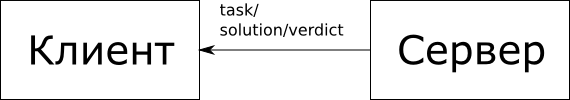
1. название файла (текст);
2. содержимое файла (массив байтов).
3. **Взаимодействие клиентов и серверов**
   1. Для аутентификации клиент отправляет запрос на сервер.



* 1. Сервер отправляет ответ о подтверждении аутентификации или отказе.



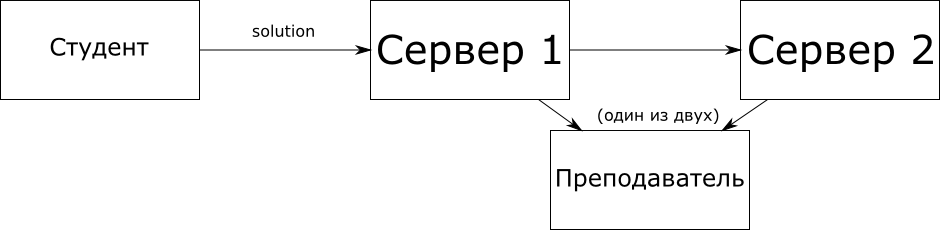
* 1. В случае успешной аутентификации сервер также отправляет новые данные, адресованные данному клиенту.



* 1. Преподаватель отправляет задание, задание пересылается другому серверу и рассылается студентам.



* 1. Студент отправляет решение, решение пересылается другому серверу, и один из серверов отправляет решение преподавателю для проверки.



* 1. Преподаватель отправляет вердикт, вердикт пересылается другому серверу, и один из серверов отправляет вердикт студенту.

