DRAFT
Version 1.0 [22.02.2006]

ПРОТОКОЛ КЛИЕНТ-СЕРВЕРНОГО ВЗАИМОДЕЙСТВИЯ «OLYMPIA»

Разработка: *OLYMPIA*™

Февраль 2006

Центр Новых Информационных Технологий
Тверской государственный университет

(4822) 36-57-43

Оглавление

1. Компоненты системы

Система автоматизации тестирования «Olympia» является клиент-серверной системой, состоящей из набора модулей, общающихся по протоколу, описанному в данном документе. В текущей версии система состоит из следующих модулей: сервер, тестер, клиент, М-клиент, администратор, рейтинг-сервер. Ниже приведено краткое описание каждого из модулей системы.

Сервер.

Сервер обеспечивает маршрутизацию всех информационных потоков между компонентами. Основная сущность, которой оперирует сервер – это *процесс тестирования*. Каждый процесс тестирования имеет свою конфигурацию, считываемую из базы данных, а также формируемую динамически при подключении или отключении клиентских компонент[1]. *Конфигурация процесса тестирования* включает в себя:

1) идентификатор тестирования (тип и номер)[2];
2) имя тестирования – произвольное название, необходимое для административных нужд;
3) флаг активности – данный атрибут существует только в базе данных и в память сервера не считывается. Он показывает, какие процессы тестирования надо загрузить в память, а какие нет;
4) флаг готовности процесса тестирования:
    a. «готов» – в настоящий момент к серверу подключены все необходимые для данного процесса тестирования компоненты и они работоспособны. Процесс тестирования начнется по расписанию;
    b. «не готов» – в настоящий момент процесс тестирования не может быть начат (или продолжен), так как не все необходимые компоненты присутствуют;
    c. «активный» – готовый процесс тестирования, который был запущен и в настоящий момент идет.
5) флаг StrictGUIDChecking – если поставлен, то все тестеры внутри одной группы возможностей[3] должны обладать одинаковым GUID'ом. В противном случае тестер не регистрируется в группе. GUID группы задается первым подключившимся в данной группе тестером;
6) сроки проведения тестирования – дата начала и дата конца. Дата конца может быть равна или меньше даты начала – в этом случае считается, что процесс тестирования никак не ограничен во времени;
7) тестовые потребности (набор строковых идентификаторов, характеризующих разрешенных тестеров) [3];
8) политика безопасности (IP адреса, пароли и кодовые слова для клиентских компонент);
9) пул тестеров, обслуживающих процесс тестирования;
10) пул клиентов, участвующих в процессе тестирования;
11) М-клиента.

Помимо обслуживания процессов тестирования, сервер также взаимодействует с административными клиентами и рейтинг-серверами. Вся информация о протекающих процессах тестирования записывается сервером в обобщенный лог. Административные клиенты управляют работой сервера, а рейтинг-серверы работают с обобщенным логом.

М-клиент (мета-клиент).

Интеллектуальная компонента системы, содержащая в себе всю логику и всю информацию, необходимую для проведения конкретных процессов тестирования. Передает серверу вопросы для отсылки клиенту, основываясь на информации о том, кто, как и когда какие

[1] под клиентскими компонентами понимаются не только клиенты системы, которые решают задачи и посылают решения на проверку, а вообще все компоненты системы тестирования за исключением сервера
[2] подробное описание идентификатора тестирования приведено в главе «Идентификатор тестирования»
[3] подробнее см. в главе «Запуск и взаимодействие компонент»

вопросы уже решил и кто какие вопросы решает в настоящий момент. Все вопросы, тесты, а также вся логика тестирования заключены в М-клиенте. Основные действия, совершаемые М-клиентом:

1) передает серверу пакет тестов для последующей передачи его тестерам;
2) передает серверу очередной вопрос для клиента;
3) получает от сервера результат проверки тестером клиентских решений;
4) получает извещение сервера о начале тестирования;
5) получает извещение сервера о завершении тестирования.

Тестер.

Основные действия и свойства тестера:

1) умеет тестировать задачи определенного типа;
2) характеризуется строкой возможностей, которую отсылает в момент регистрации;
3) получает от сервера пакет тестов;
4) принимает от тестера задачу и возвращает результат тестирования.

Тестер не привязан к какому-либо конкретному процессу тестирования. Он умеет проверять любые задачи в рамках одного типа тестирования. Однако, одновременно тестер может обслуживать только один процесс тестирования.

Клиент.

Приложение, с которым непосредственно работает пользователь. Функциональность клиента в общем случае зависит от задачи, выполняемой процессом тестирования. Основные действия, выполняемые клиентом:

1) принимает от сервера очередной вопрос;
2) посылает серверу ответ на вопрос. Ответ характеризуется строкой потребностей, по которой сервер понимает, какому тестеру его отослать.

Администратор.

Приложение, позволяющее управлять работой сервера. В текущей версии протокола единственной административной командой является команда перезагрузки конфигурации сервером.

Рейтинг-сервер.

Содержит локальную копию лога, которую он передает любому клиенту извне. Форма, в которой он предоставляет лог, произвольна. Рейтинг-сервер необходим для того, чтобы защитить сервер от сетевых атак и чтобы вынести логику формирования и выдачи информации о тестировании из сервера. Например, в случае олимпиад по программированию, проводимых по правилам ACM, рейтинг-сервер должен выдавать результаты олимпиады в виде конкретно заданной турнирной таблицы и должен уметь замораживать рейтинг за некоторое время до конца олимпиады.

2. Идентификатор тестирования

Процессы тестирования описываются *типом* и *номером тестирования.*

Тип тестирования – уникальный идентификатор, характеризующий множество клиентских компонент (тестеров, м-клиентов, клиентов, рейтинг-серверов), которые умеют общаться друг с другом. Компоненты разных типов в общем случае работать друг с другом не могут, т.е., например, решение олимпиадной задачи по программированию не может быть проверено тестером для практикумов по теории вероятностей. Одно из важных свойств сервера – возможность одновременного независимого проведения тестирований разных типов.

Помимо типа тестирования каждый конкретный процесс тестирования характеризуется номером тестирования, необходимого для обеспечения возможности проведения одновременно нескольких тестирований одного типа, а также для более удобного хронологического идентифицирования тестирований.

Тип тестирования и номер тестирования вместе составляют *идентификатор тестирования*, который однозначно идентифицирует определенный *процесс тестирования*.

Тип тестирования задается последовательностью прописных и заглавных латинских букв, символа подчеркивания и цифр. Начинаться тип тестирования должен либо с буквы латинского алфавита, либо с символа подчеркивания. Номер тестирования – неотрицательное целое число. Идентификатор тестирования состоит из типа тестирования, символа точки и номера тестирования. Например:

olymp.0

Клиент, тестер и М-клиент могут одновременно участвовать только в одном процессе тестирования. Более того, один произвольный процесс тестирования может обслуживаться только одним М-клиентом. Так как вся логика тестирования, а также информация, необходимая для непосредственного проведения процесса тестирования (вопросы, тесты) находятся у М-клиента, то у каждого М-клиента есть конкретно заданный список идентификаторов тех процессов тестирования, которые он может проводить.

При подключении к серверу клиент передает ему идентификатор процесса тестирования, высказывая тем самым свое желание в нем участвовать. Тестер при подключении к серверу посылает лишь тип тестирования, при этом сервер выбирает подходящий процесс тестирования и посылает его идентификатор обратно тестеру (либо сообщает, что в настоящий момент в услугах данного тестера не нуждается). М-клиент при подключении к серверу посылает несколько идентификаторов тестирования, указывая тем самым список тех процессов тестирования, которые могут быть проведены данным М-клиентом. Сервер выбирает из них один, который еще никаким другим М-клиентом не обслуживается, и посылает идентификатор данного процесса тестирования М-клиенту.

После того, как процесс тестирования заканчивается, сервер рассылает соответствующим тестерам и М-клиенту сообщение 112 (*Service Unneeded*) и переводит их в пул свободных компонент. Там они находятся в течение некоторого времени, пока не произойдет одно из двух событий:

1. Появится подходящий процесс тестирования. В этом случае сервер регистрирует в нем соответствующие компоненты и отсылает им ответ 200 (*Logged In*).
2. Пройдет определенный промежуток времени, за который подходящий процесс тестирования так и не появится. В этом случае сервер закрывает канал связи.

Рейтинг-сервер и административный клиент при подключении к серверу идентификатора тестирования не указывают. Административный клиент управляет не только частными процессами тестирования, но и общим поведением сервера. Рейтинг-сервер работает только с обобщенным логом, посылая запросы на выдачу лога и указывания в этих запросах идентификатор того процесса тестирования, чей лог нужен. На выдачу лога никак не влияет тот факт, находится ли данный процесс тестирования в памяти или нет. Поэтому ни административный клиент, ни рейтинг-сервер жестко не привязаны к тем процессам тестирования, которые в настоящий момент обслуживает сервер, однако выдача лога рейтинг-серверу может быть ограничена действующей политикой безопасности.

3. Запуск и взаимодействие компонент

Загрузка сервера

Сервер, запускаясь, считывает свою стартовую конфигурацию из конфигурационного файла. В этом файле, помимо прочей информации, также содержатся настройки доступа к базе данных сервера. Считав и обработав всю информацию из конфигурационного файла, сервер пытается подключиться к базе данных. Если ему это не удается, то он выводит сообщение об ошибке и завершает работу. Если ему это удается, то он устанавливает подключение к базе данных и считывает из нее свою основную конфигурацию. Данная конфигурация состоит из конфигураций всех процессов тестирования, дата завершения которых еще не настала и которые помечены как «активные» (поставлен флаг активности – не надо путать с состоянием процесса тестирования).

Помимо конфигурации сервер обладает также «состоянием», определяемым следующими списками:

1) списком всех подключенных компонент (тестеров, клиентов, М-клиентов и т.д.);
2) списком всех процессов тестирования.

Конфигурацию сервера можно менять во время работы сервера путем изменения соответствующей информации в базе данных и посылки серверу сигнала перезагрузки. Сервер считает новые настройки из базы данных и предпримет соответствующие действия.

Подключение клиентских компонент к серверу

Тестер.

Тестер, регистрируясь в системе, отправляет серверу три ключевых поля – тип тестирования, обслуживаемый данным тестером, GUID – уникальный код тестера, и строку возможностей, характеризующую те возможности по проверке решений, которыми обладает тестер. Например, в случае олимпиад по программированию в данную строку возможностей может входить перечень тех компиляторов, которые поддерживает тестер, а также та платформа, на которой работает тестер и, соответственно, на которой он может компилировать решения олимпиадных задач. Данная строка возможностей состоит из строковых идентификаторов, разделенных запятой.

Для определения того, может ли тестер быть зарегистрирован в данном процессе тестирования или нет, а также определения наличия всех необходимых для начала процесса тестирования тестеров, сервер использует *тестовые потребности*, записанные в конфигурации процесса тестирования. Тестовые потребности представляют собой набор строчек, каждая из которых есть множество текстовых идентификаторов, разделенных запятой:

$$id_{11}[*],id_{12}[*],\ldots,id_{1n_1}[*]$$
$$id_{21}[*],id_{22}[*],\ldots,id_{2n_2}[*]$$
$$\ldots$$
$$id_{m1}[*],id_{m2}[*],\ldots,id_{mn_m}[*]$$

[*] – означает, что после каждого идентификатора может опционально идти символ *. Во время подключения тестера, сервер анализирует строку возможностей тестера на соответствие тестовым возможностям всех готовых и неготовых процессов тестирования.

Будем говорить, что строка возможностей соответствует тестовым потребностям некоторого процесса тестирования, если:

1. найдется хотя бы одна строчка в тестовых потребностях процесса тестирования такая, что все идентификаторы в ней, непомеченные символом *, встречаются в строке возможностей тестера;
2. все оставшиеся идентификаторы строки возможностей тестера встречаются среди идентификаторов данной строчки тестовых потребностей, помеченных символом *.

Если строка возможностей тестера соответствует тестовым потребностям некоторого процесса тестирования, то данный процесс тестирования называется подходящим для тестера. Следует заметить, что строка возможностей тестера проверяется на соответствие тестовым потребностям процесса тестирования только в том случае, если политика безопасности процесса

тестирования, либо глобальная политика безопасности (в случае, когда процесс тестирования не имеет своей локальной политики безопасности) позволяет подключение данного тестера к данному процессу тестирования и, очевидно, если тип процесса тестирования совпадает с типом тестера.

Затем из списка подходящих процессов тестирования выбирается один – тот, который начнется раньше всех, и в пул его тестеров помещается рассматриваемый тестер. Если два или более процессов тестирования имеют одинаковую дату начала и начинаются раньше всех остальных, то выбирается тот, у кого в настоящий момент зарегистрировано меньше всего тестеров. По умолчанию сервер не рассматривает активные процессы тестирования, которые идут в настоящий момент.

Для того, чтобы процесс тестирования смог начаться, необходимо, чтобы все его тестовые потребности были удовлетворены. Для этого необходимо, чтобы в его пуле тестеров находились тестеры, покрывающие своими строчками возможностей все строчки тестовых потребностей данного тестирования. Строчка тестовых потребностей тестирования считается полностью покрытой, если в пуле тестеров данного процесса тестирования найдется такое подмножество тестеров, объединив[1] строчки возможностей которых мы получим строку, полностью совпадающую[2] с данной строчкой тестовых потребностей.

Например, пусть процесс тестирования имеет следующие тестовые потребности:

c*,pascal*,java*,unix
c*,pascal*,java*,windows

Подключение следующего множества тестеров не сможет покрыть данные потребности:

Тестер 1: c,java,windows
Тестер 2: pascal,unix
Тестер 3: java,c,unix

Однако при подключении четвертого тестера, например – c,pascal,windows, данное множество, состоящее из четырех тестеров, сможет полностью покрыть тестовые потребности тестирования, так как:

1) строка №1 тестовых потребностей: c*,pascal*,java*,unix – покрывается тестерами №2 и №3;
2) строка №2 тестовых потребностей: c*,pascal*,java*,windows – покрывается тестерами №1 и №4.

Стоит отметить, что если к данной системе подключится тестер со следующей строкой возможностей: c,pascal,windows,unix – то приведенный выше процесс тестирования не будет для него подходящим, так как нет ни одной строчки в его тестовых потребностях, содержащей одновременно unix и windows.

Подключенному тестеру сервер высылает уведомление о регистрации в системе вместе с идентификатором того процесса тестирования, в котором он был зарегистрирован. Далее тестер может послать запрос на получение тестового пакета данного процесса тестирования.

Если еще нет ни одного процесса тестирования, в котором мог бы принять участие тестер, или текущий обслуживаемый процесс тестирования закончился сервер посылает тестеру ответ 112 (*Service Unneeded*) и переводит его в пул свободных тестеров, из которого он будет удален, т.е. отключен, через определенный промежуток времени, если за это время не появится подходящий процесс тестирования.

GUID характеризует совместимость (одинаковость) тестеров на уровне бинарного кода. Каждый процесс тестирования имеет флаг StrictGUIDChecking, который может быть поставлен, если необходимо, чтобы внутри каждой группы возможностей все тестеры были абсолютно одинаковыми.

[1] объединение здесь понимается в смысле обычной операции объединения на множествах
[2] под совпадением подразумевается эквивалентность данных множеств

Клиент.

Клиент, подключаясь, посылает серверу идентификатор того процесса тестирования, в котором он хочет принять участие, а также кодовое слово. Север проверяет, соответствует ли клиент политике безопасности того процесса тестирования, к которому он хочет подключиться, либо глобальной политике безопасности. Если соответствует, то клиент помещается в список клиентов данного процесса тестирования.

Когда процесс тестирования начинается, сервер рассылает соответствующие уведомления всем клиентам. После уведомления клиент запрашивает у сервера первый вопрос.

Если в системе нет процесса тестирования, к которому подключается клиент, то сервер отключает данного клиента.

М-клиент.

В момент подключения М-клиент посылает серверу список идентификаторов процессов тестирования, которые он может обслуживать. Сервер проверяет данного М-клиента на соответствие политикам безопасности. При этом сервер рассматривает только те процессы тестирования, которые еще не имеют подключенного к ним М-клиента. После этого сервер выбирает один из подходящих процессов тестирования, имеющего наиболее раннюю дату начала, и посылает его идентификатор М-клиенту.

Для данного процесса тестирования М-клиент высылает серверу тестовый пакет и сообщает о своей готовности принимать и обрабатывать запросы.

После того, как процесс тестирования заканчивается, или если при подключении не находится подходящего процесса тестирования, то сервер помещает М-клиента в пул свободных М-клиентов и отсылает ему сообщение 112 (*Service Unneeded*). Если в течение определенного промежутка времени появляется подходящий процесс тестирования, сервер посылает М-клиенту ответ 200 (*Logged In*) с идентификатором данного процесса тестирования. В противном случае по истечению заданного интервала времени сервер отключает М-клиента.

Административный клиент.

Административный клиент при подключении отсылает пароль. При условии, что данный пароль корректен и административный клиент находится на машине с IP адресом, прописанным в базе данных, сервер устанавливает соединение.

Следует заметить, что административному клиенту не надо посылать серверу в момент регистрации никаких идентификаторов процессов тестирования, так как в текущей версии протокола административный клиент управляет общим поведением сервера и не привязан к какому-либо конкретному процессу тестирования.

Рейтинг-сервер.

Рейтинг-сервер, как и административный клиент, при подключении не указывает никаких идентификаторов процессов тестирования. Тем не менее, каждый запрос на выдачу лога должен содержать идентификатор того процесса тестирования, чей лог запрашивается.

Если адрес машины, с которой подключается рейтинг-сервер, не прописан в базе данных ни в глобальной политике безопасности, ни в локальных сервер не открывает Р-канала. Если адрес находится в глобальной политике безопасности, то сервер позволяет рейтинг-серверу запрашивать лог любого процесса тестирования, за исключением случая, когда в политике безопасности конкретного процесса тестирования прописан хотя бы один IP адрес рейтинг-сервера – в этом случае лог данного процесса тестирования сервер будет выдавать рейтинг-серверам, находящимся только на данных IP адресах.

Административным клиентам выдается лог любого процесса тестирования, независимо от политик безопасности. М-клиентам выдается лог того процесса тестирования, в котором он сейчас зарегистрирован, независимо от политик безопасности.

Взаимодействие компонент

В рамках одного конкретного процесса тестирования взаимодействие компонент системы происходит следующим образом:

1. В момент начала тестирования сервер рассылает соответствующим клиентским компонентам уведомление о начале процесса тестирования.
2. Каждый клиент посылает серверу запрос: «Дай мне задачу». Сервер перенаправляет этот запрос М-клиенту.
3. Получив от М-клиента вопрос для i-го клиента, сервер регистрирует его в обобщенном логе и отсылает клиенту.
4. Когда j-тый клиент присылает серверу ответ на вопрос, сервер регистрирует ответ в обобщенном логе, смотрит на требования, которые клиент прислал вместе с ответом, находит нужную группу тестеров и ставит ответ в очередь в эту группу на проверку.
5. Когда тестер присылает результат проверки ответа на вопрос серверу, сервер регистрирует данный результат в обобщенном логе и отсылает его как М-клиенту, так и самому клиенту.
6. Затем клиент спрашивает у сервера следующий вопрос. Сервер адресует запрос М-клиенту. Если вопрос есть, то процесс тестирования продолжается. Если М-клиент присылает сообщение, что вопросов больше нет, сервер посылает клиенту уведомление об окончании процесса тестирования.

Алгоритм нахождения нужной группы тестеров следующий – сервер просматривает пул тестеров данного процесса. Все тестеры в этом пуле разбиты на группы, согласно своим возможностям, декларированным в момент подключения. Тестер сравнивает строку потребностей ответа клиента с возможностями каждой группы, выбирает те группы, строки возможностей которых соответствуют строке потребностей клиента. Среди данных отобранных групп выбирается группа с наименьшим коэффициентом загрузки, который вычисляется как количество задач в очереди на данную группу, деленное на количество тестеров в данной группе. В очередь к данной группе и ставится задача.

4. Обобщенный лог

В обобщенном логе сервер регистрирует все сообщения, пересылаемые между компонентами - участниками тестирования (клиент, М-клиент, тестер). Данный лог называется обобщенным, потому что его формат не зависит от конкретного типа тестирования.

С точки зрения сервера в рамках одного процесса тестирования есть три сущности:

- вопрос - передается сервером от М-клиента клиенту;
- ответ на вопрос – передается сервером от клиента тестеру;
- результат проверки ответа – передается сервером от тестера М-клиенту и клиенту.

Все эти три сущности, вместе со временем их поступления в систему, а также клиентом-источником и клиентом-получателем регистрируются в обобщенном логе.

Запросы на выдачу лога выбирают из обобщенного лога только вопросы и результаты проверки ответов, в то время как ответы клиентов в рейтинг-сервер не попадают.

Обобщенный лог имеет следующие поля:

1. id - идентификатор записи
2. TId - идентификатор процесса тестирования
3. fromType – тип источника (client, meta, tester)
4. fromName – имя источника (кодовое слово для клиента, GUID для тестера, пусто для М-клиента)
5. toType – тип приемника (client, meta, tester)
6. toName – имя приемника (кодовое слово для клиента, GUID для тестера, пусто для М-клиента)
7. dateEntered – дата с точностью до секунды поступления сообщения в систему от источника
8. dateLeft – дата с точностью до секунды ухода сообщения из системы приемнику
9. body – тело сообщения

5. Каналы связи

Общение между клиентской компонентой и сервером происходит через виртуальный канал связи. Канал связи между клиентом и сервером устанавливается по инициативе клиента после открытия сетевого соединения. Сетевое соединение открывается только в том случае, если клиент удовлетворяет всем политикам безопасности (корректный IP адрес, корректный пароль и т.д.). Таким образом, подключение к серверу с посторонних IP адресов невозможно. После установки сетевого соединения, клиент может послать запрос LOGIN на открытие одного из каналов связи. Если по прошествии определенного времени клиент не открывает никакого канала связи, то сервер отключает его.

Каналы связи бывают следующих типов:

1. виртуальный канал для общения клиента-участника и сервера – КУ-канал;
2. виртуальный канал для общения тестера и сервера – Т-канал;
3. виртуальный канал для общения М-клиента и сервера – М-канал;
4. виртуальный канал для общения администратора и сервера – А-канал;
5. виртуальный канал для передачи обобщенного лога – Р-канал.

Р-канал автоматически открывается поверх А-канала и М-канала и не требует отсылки дополнительного запроса LOGIN. В остальных случаях открытие второго канала связи поверх другого невозможно. При запросе LOGIN в одном из каналов связи, сервер возвращает отклик 401 *(Method Not Allowed)*.

Основная идея заключается в том, что каждому каналу соответствует свой круг клиентских запросов, которые могут быть приняты и обработаны сервером, только если они были посланы через соответствующий канал, при этом все запросы для каждого типа канала (кроме запроса LOGOUT) не пересекаются (следует отметить, что Р-канал открывается автоматически вместе с М- и А-каналами). Например, перезагрузить сервер можно только по А-каналу связи, а запросить тестовый пакет по Т-каналу. Даже административный канал связи не обладает всей полнотой запросов, так как это привело бы к излишней загруженности администратора ненужной функциональностью. По сути каналы связи позволяют в одном протоколе совместить сразу пять разных протоколов – по одному на каждый тип клиента.

Есть только один свободный запрос, который можно отослать вне какого-либо канала – запрос LOGIN.

Как было сказано ранее, канал связи устанавливается клиентом с помощью запроса серверу LOGIN. Параметр запроса указывает на тип создаваемого канала. Основными условиями открытия виртуального канала связи служат:

1. для КУ-каналов – клиент должен находиться на машине с IP адресом, зарегистрированным за одним из клиентов. Помимо IP адресов в базе данных могут храниться адреса целых подсетей, в которых работают клиенты. Помимо этого, заголовок Password запроса должен содержать кодовое слово, принадлежащее данному клиенту. Кодовое слово необходимо М-клиенту для идентификации пользователя. В случае, если клиент отстранен от тестирования, сервер не открывает канал.
2. для Т-каналов – клиент должен находиться на машине с IP адресом, прописанным в базе данных как IP адрес тестирующей машины. Тестер, регистрируясь в системе, отправляет серверу три ключевых поля – тип тестирования, обслуживаемый данным тестером, GUID – уникальный код тестера и строку возможностей, характеризующую те возможности по проверке решений, которыми обладает тестер[1]. В случае невозможности регистрации тестера, вызванной нарушением политики

[1] Более подробно процесс регистрации тестера и других клиентских компонент описан в главе «Запуск и взаимодействие компонент».

безопасности, сервер возвращает ошибку 400 (*Forbidden*) и не открывает канал. Если невозможность регистрации вызвана отсутствием в настоящий момент подходящих процессов тестирования, то сервер посылает ему ответ 112 (*Service Unneeded*) и помещает тестера в специальный пул свободных тестеров, где он ждет в течение некоторого времени появления подходящего процесса тестирования. Если такой процесс тестирования так и не появляется, сервер отключает данного тестера, посылая ему ответ 201 (*Bye*).

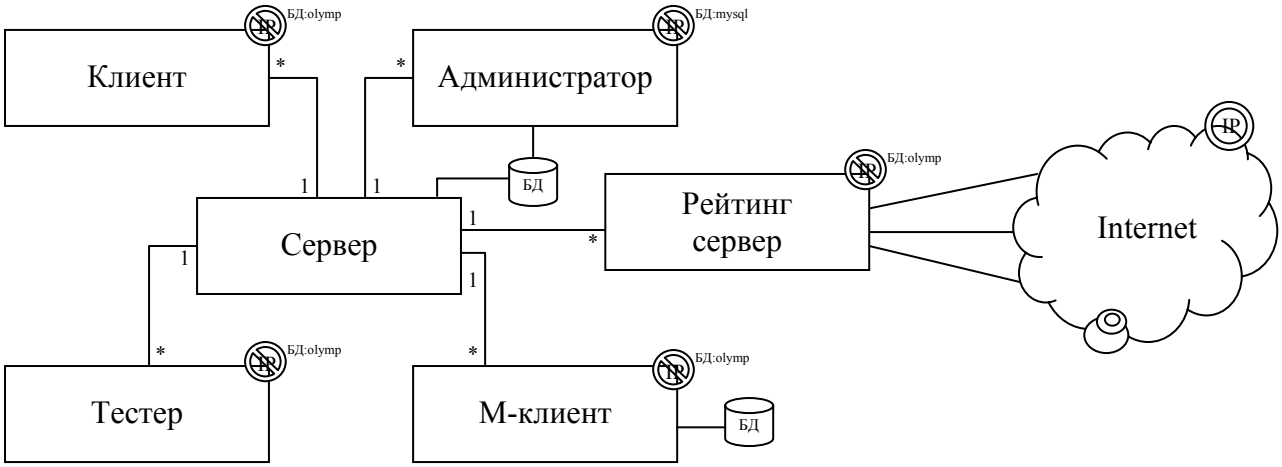
3. для А-каналов - клиент должен находиться на машине с IP адресом, прописанным в системной базе данных (mysql), как IP адрес администраторской машины, и заголовок Password запроса должен содержать администраторский пароль, совпадающий с паролем, хранящимся в той же системной базе данных (mysql).
4. для М-клиентов канал открывается, если клиент находиться на машине с IP адресом, прописанным в базе данных, как IP адрес М-клиента, а также если существуют в настоящий момент такие процессы тестирования, которые могут быть обслужены данным М-клиентом. В случае невозможности регистрации М-клиента, вызванной нарушением политики безопасности, сервер возвращает ошибку 400 (*Forbidden*) и не открывает канал. Если невозможность регистрации вызвана отсутствием в настоящий момент подходящих процессов тестирования, то сервер посылает ему ответ 112 (*Service Unneeded*) и помещает М-клиента в специальный пул свободных М-клиентов, где он ждет в течение некоторого времени появления подходящего процесса тестирования. Если такой процесс тестирования так и не появляется, сервер отключает данного М-клиента, посылая ему ответ 201 (*Bye*).
5. для Р-каналов - клиент должен находиться на машине с IP адресом, прописанным в базе данных, как IP адрес М-клиента, администраторской машины или рейтинг-сервера.

В случае, если клиент попытается сделать запрос, запрещенный в данном канале связи, сервер вернет ошибку несоответствия запроса каналу связи 401 (*Method Not Allowed*). Если же клиент попытается установить канал связи, не соответствующий его (клиента) IP адресу (например, кодовое слово или пароль не соответствуют действительным – для А- и КУ-каналов), то сервер возвращает ответ 400 (*Forbidden*) с текстовым описанием ошибки в заголовке Message.

Канал связи может быть закрыт клиентом либо посредством запроса LOGOUT, либо при закрытии сетевого соединения.

6. Состояния каналов

Общую схему клиент-серверного взаимодействия можно описать следующей диаграммой:



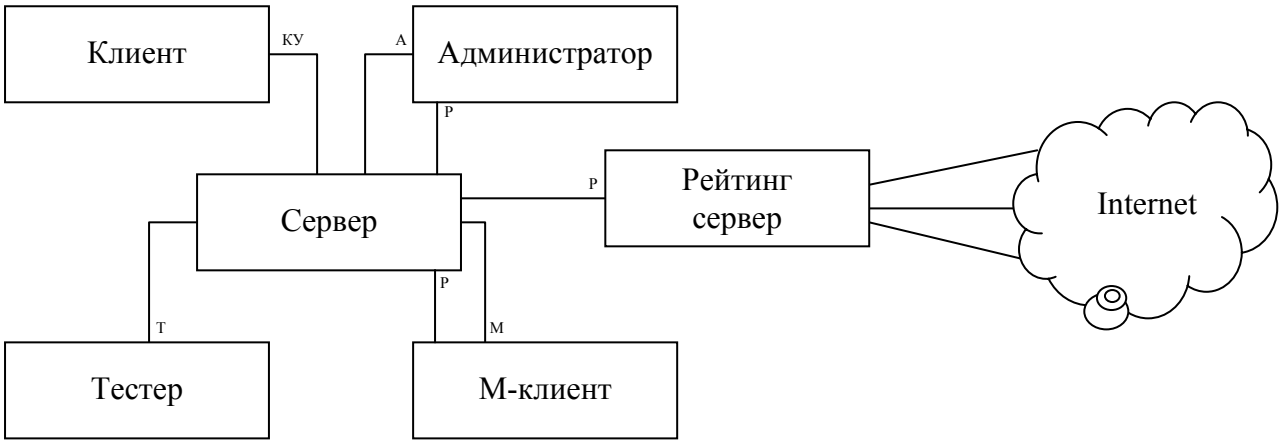
где:

1. IP – доступ разрешен с любого IP адреса
2. БД:olymp - доступ разрешен только с тех IP адресов, что прописаны в базе данных olymp
3. БД:mysql - доступ разрешен только с тех IP адресов, что прописаны в базе данных mysql

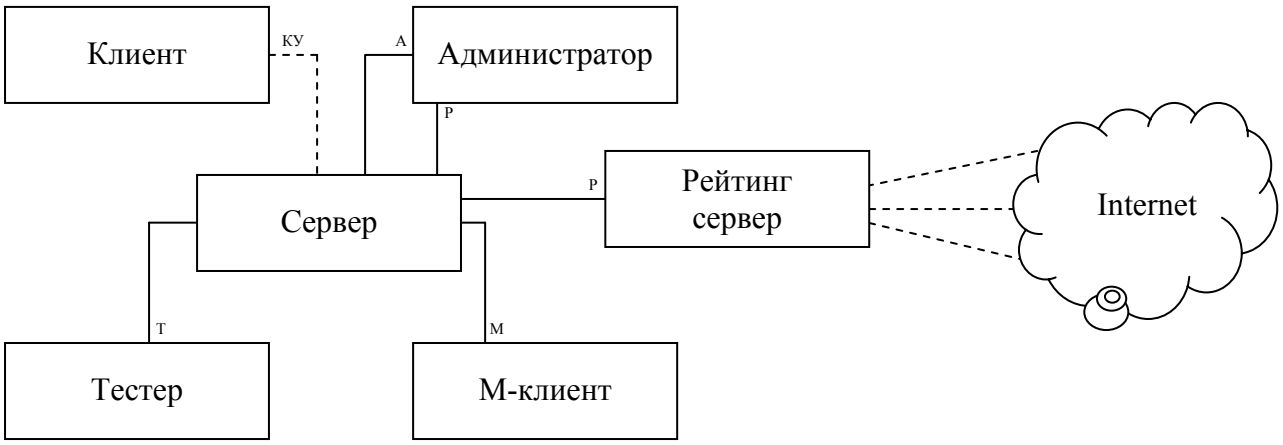
Таким образом, напрямую с базой данных могут общаться только сервер и администратор. Причем IP адрес и пароли администраторов прописаны в системной базе данных mysql, что предотвращает несанкционированный доступ в БД администраторов с произвольной машины.

В рамках одного конкретного процесса тестирования, сервер может быть в одном из следующих состояний:

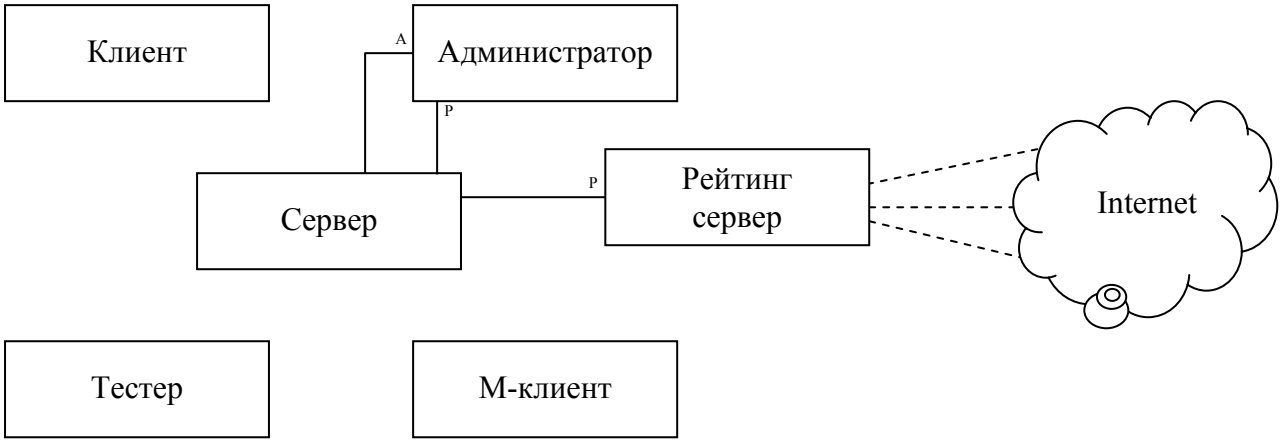
1) тестирование идет (КУ, М, А, Т, Р – соответствующие каналы передачи данных, линии без подписей – прямой доступ)



2) тестирование не идет, статус – «готово» или «не готово»



3) тестирования нет (прошло)



Линии пунктиром обозначают замороженный канал. Клиенты не отключаются от сервера, но ни один запрос не принимается на обработку.

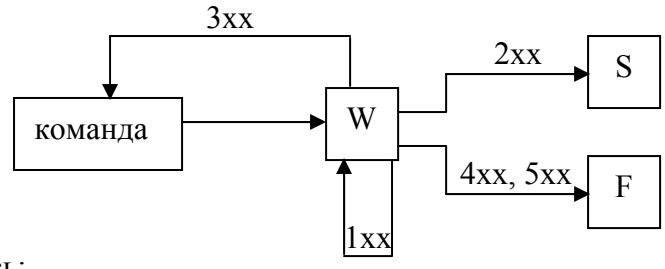
7. Принципы кодирования откликов сервера

Код отклика сервера является трехзначным числом, в котором первая цифра идентифицирует тип отклика, а остальные две образуют порядковый номер отклика.

Существует пять значений для первой цифры кода отклика:

- 1yz – позитивный предварительный отклик (информационное сообщение сервера)
  Запрос принят и успешно выполнен, но прежде чем отсылать новый запрос, клиент должен дождаться еще одного отклика от сервера.
- 2yz – позитивный заключительный отклик
  Запрос был успешно выполнен. Может быть инициирован новый запрос.
- 3yz – позитивный промежуточный отклик
  Запрос был принят, но его исполнение временно приостановлено в ожидании дополнительной информации от клиента. Клиент должен послать следующий запрос с необходимой серверу информацией. Данный тип откликов используется в составных запросах, когда одна команда выполняется посредством некоторой цепочки запросов и ответов.
- 4yz – временно-отрицательный заключительный отклик (ошибка клиента)
  Запрос был отвергнут, и запрашиваемая команда, соответственно, не была выполнена. Тем не менее, ошибка, из-за которой запрос не был принят на исполнение, не является фатальной, и клиент может повторить запрос, изменив, быть может, некоторые его параметры или подождав некоторое время. Точное значение “временности” зависит от конкретного запроса.
- 5yz – перманентно-отрицательный заключительный отклик (ошибка сервера)
  Запрос был отвергнут, и запрашиваемая команда, соответственно, не была выполнена. Клиент не должен посылать тот же самый запрос, так как его обработка невозможна из-за неустранимой ошибки сервера (такой ошибки, которая не может быть устранена сервером самостоятельно, и сервер не имеет никакой дополнительной информации о том, когда и как она будет устранена).

Взаимозависимость откликов сервера можно изобразить на следующей диаграмме:



Здесь:

- команда – команда, посылаемая клиентом серверу на выполнение
- W – состояние ожидание отклика сервера
- S – успешное выполнение команды
- F – неуспешное выполнение команды

8. Сводные таблицы запросов и ответов

Запросы	*	КУ	М	Т	А	Р
LOGIN	✔					
LOGOUT		✔	✔	✔	✔	✔
C-READY		✔				
C-DONE		✔				
M-READY			✔			
M-DONE			✔			
TTP			✔			
T-READY				✔		
T-DONE				✔		
GTP				✔		
INIT					✔	
RATING						✔
RATING-PART						✔
PROFILE						✔

Расшифровка обозначений:

- * - запрос возможен вне какого-либо канала
- КУ – запрос возможен в КУ-канале
- М – запрос возможен в М-канале
- А – запрос возможен в А-канале
- Т – запрос возможен в Т-канале
- Р – запрос возможен в Р-канале

Ответы
100 Wait For Beginning
101 Answer Accepted
102 Registered
103 Testing Not Ready
104 Queued
112 Service Unneeded
200 Logged In
201 Bye
202 Result Of Testing
203 Test Packet
204 Result Accepted
205 OK
206 Full Log
207 Part Of Log
208 Log Not Changed
209 Testing Started
210 Profile
211 Testing Is Over
213 Question Accepted
220 <имя сервера> at <имя хоста>
300 Reload Test Packet
301 Answer
302 Question
303 Request For Question
400 Forbidden
401 Method Not Allowed
402 Client Disqualified
403 Length Required
404 Bad Request
410 Wrong Test Id
500 Internal Server Error
501OLYMPIA Version Not Supported

Если запрос не соответствует каналу связи, сервер возвращает сообщение об ошибке 401 *(Method Not Allowed)*. Если же запрос (кроме LOGIN) используется вне какого-либо канала, то сервер возвращает отклик 400 *(Forbidden)*.

9. Запросы

9.1. Синтаксис запроса

Запрос имеет следующий синтаксис в БНФ-нотации:

запрос := команда <CRLF> [заголовки] <CRLF> [тело]
команда := <команда> [<SP> <параметр>] <SP> <протокол>/<версия>
<команда> := LOGIN | LOGOUT | C-READY | C-DONE | M-READY | M-DONE |
TTP | T-READY | T-DONE | GTP | INIT | RATING |
RATING-PART | PROFILE
<параметр> := client | tester | admin | meta | rating | with-last-timestamp | finished
<протокол> := OLYMPIA
<версия> := [0-9]+ . [0-9]+
заголовки := заголовок <CRLF> | заголовок <CRLF> заголовки
заголовок := <Имя заголовка>: <значение>
<Имя заголовка> := Password | TId | TType | GUID | Content-Length | Possibilities |
Reason| Requirements | Error | From
тело := {текст, предусмотренный запросом. Длина тела указана в Content-Length}
<SP> := _ | \t
<CRLF> := \n

Длина строки команды и заголовков не может превосходить 1024 символов. Количество заголовков также не может превосходить 1024. Если запрос подразумевает наличие тела, то среди заголовков обязательно должен присутствовать заголовок Content-Length с длиной тела в байтах.

Регистр команд, параметров, заголовков, а также названия протокола не важен. Концом заголовка запроса (включающего строку команды и опциональный блок заголовков) является пустая строка. Команда представляет собой последовательность из латинских букв и дефиса. Порядок заголовков не имеет значения, имя заголовка отделяется от его значения двоеточием. Заголовок, а также строка команды не должны содержать начальных пробелов.

9.2. Описание запросов

Во всех диаграммах состояния, приводимых для запросов, используются следующие обозначения:

- B – начало команды
- W – состояние ожидания клиентом ответа от сервера
- S – успешное завершение команды
- E – неуспешное завершение команды
- C – состояние ожидания сервером запроса от клиента. Встречается в изображении диаграмм составных команд

LOGIN

Назначение: открытие виртуального канала передачи данных.

Параметры:

- **client** – открытие КУ-канала между сервером и клиентом-участником тестирования
- **meta** – открытие М-канала между сервером и М-клиентом
- **tester** – открытие Т-канала между сервером и тестером
- **admin** – открытие А-канала между сервером и администратором
- **rating** – открытие Р-канала между сервером и рейтинг-сервером

Заголовки:

- Password: кодовое слово клиента или пароль администратора, в зависимости от того, какой канал открывается. Заголовок посылается клиентом в том случае, если заданы параметры **client** или **admin** – в остальных случаях сервер игнорирует данный заголовок.
- TId: идентификатор процесса тестирования. КУ-клиент посылает в данном заголовке идентификатор того процесса тестирования, в котором он хочет принять участие. М-клиент посылает серверу список идентификаторов тестирования, разделенных символом ‘,’ – тех процессов тестирования, которые могут быть обслужены данным М-клиентом.
- TType: тип тестирования, посылаемый тестером при открытии Т-канала для указания того типа тестирования, который может обслуживаться данным тестером.
- GUID: уникальный идентификатор тестера, посылаемый им при открытии Т-канала. Предполагается, что два тестера, имеющих одинаковый GUID, совпадают с точностью до исполняемых файлов (исполняемые файлы данных модулей совпадают побайтово).
- Possibilities: строка возможностей, посылаемая тестером при открытии Т-канала, характеризующая тестовые возможности модуля. Используется сервером для выбора тех процессов тестирования, в которых тестер может участвовать и для создания групп тестеров, различающихся своими возможностями.

Допустимые каналы: вне какого-либо канала.

Описание:

Общение с сервером может происходить только через защищенный канал передачи данных. Каналов связи бывает пять – КУ-канал, М-канал, Т-канал, А-канал и Р-канал (подробнее см. в разделе «Каналы связи»). Установление канала связи происходит посредством запроса LOGIN, причем его параметр указывает на то, какой именно канал клиент хочет установить. При открытии М- и А-каналов, Р-канал устанавливается автоматически. Рейтинг-сервер должен явно указать параметр rating для того, чтобы сервер открыл Р-канал.

Сервер возвращает отклик 400 *(Forbidden)* в следующих случаях:

- если производится открытие канала, не соответствующего IP адресу;
- если делается запрос на открытие А-канала, но пароль не совпадает с реальным паролем администратора;
- если производится открытие КУ-канала с разрешенного IP адреса, но кодовое слово клиента не соответствует действительному кодовому слову, которое прописано в базе данных.

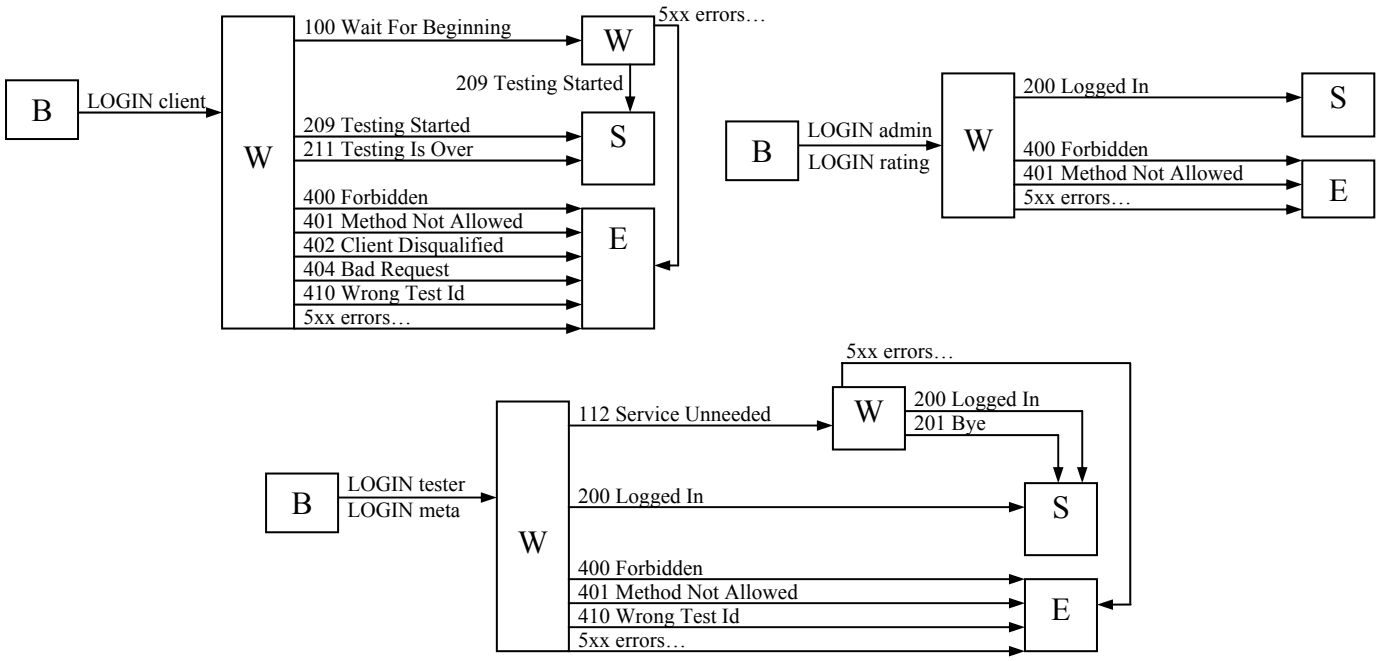
Если же делается попытка установить КУ-канал, а клиент-участник является дисквалифицированным, то сервер возвращает код ошибки 402 *(Client Disqualified)*. Если значение заголовка TId не соответствует ни одному номеру процесса тестирования, загруженного сервером, то сервер возвращает ответ 410 (*Wrong Test Id*).

В случае невозможности регистрации тестера или М-клиента, вызванной нарушением политики безопасности, сервер возвращает ошибку 400 (*Forbidden*) и не открывает канал. Если невозможность регистрации вызвана отсутствием в настоящий момент подходящих процессов тестирования, то сервер посылает ему ответ 112 (*Service Unneeded*) и помещает тестера или М-клиента в специальный пул свободных компонент, где он ждет в течение некоторого времени появления подходящего процесса тестирования. Если такой процесс тестирования так и не появляется, сервер отключает данную компоненту, посылая ей ответ 201 (*Bye*).

Если открытие канала происходит успешно, то ответ сервера зависит от типа открываемого канала:

- если открыт КУ- канал, то:
    - если тестирование еще не началось, сервер посылает клиенту ответ 100 *(Wait For Beginning)*. Затем, когда тестирование начинается, сервер посылает клиенту сообщение 209 (*Testing Started);*
    - если тестирование началось, сервер посылает 209 (*Testing Started);*
    - если тестирование закончилось, сервер посылает 211 (*Testing Is Over*);
- если открыт А-, Р-, М- или Т-канал, то сервер посылает 200 (*Logged In)*. Для М-клиента и тестера сервер вместе с ответом посылает идентификатор того процесса тестирования, в котором он зарегистрировал данный модуль.

Диаграммы состояний:



LOGOUT

Назначение: закрытие виртуального канала передачи данных.

Параметры: нет.

Заголовки:

- Reason: причина закрытия канала.

Допустимые каналы: любой канал.

Описание:

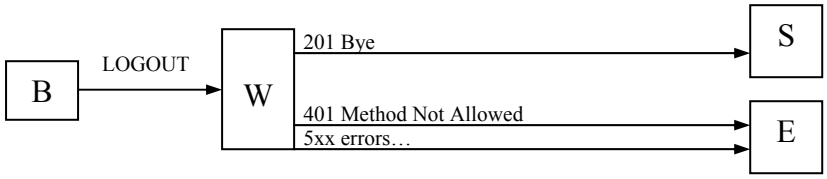
Клиент может закрыть канал с сервером по своему желанию, послав ему запрос LOGOUT. Данный запрос не имеет параметров, но может иметь заголовок Reason, в котором он указывает причину закрытия канала. Это полезно в случае, если, например, М-клиент, послав очередной запрос серверу, в ответ получает некорректные данные – в этом случае он может отключиться от сервера, указав при этом причину отключения в заголовке Reason. В случае успешного закрытия канала, сервер посылает ответ 201 *(Bye)* и закрывает сетевое соединение.

У ответа сервера 201 (*Bye*) имеется опциональное поле Reason, в котором сервер также может указать причину закрытия канала, если он является инициатором его закрытия.

В случае, если клиент посылает данную команду при закрытом канале, сервер возвращает 401 *(Method Not Allowed)*.

Использование команды LOGOUT для закрытия канала передачи данных не является обязательным.

Диаграмма состояний:



C-READY

Назначение: сообщает серверу о готовности КУ-клиента принять очередной вопрос.

Параметры: нет.

Заголовки: нет.

Допустимые каналы: КУ.

Описание:

Каждый клиент самостоятельно запрашивает у сервера очередной вопрос, отсылая ему запрос C-READY. Запрос C-READY является первым запросом в составной команде C-READY – C-DONE.

Общая схема выглядит следующим образом. Когда клиент готов принять первый вопрос (например, после получения извещения сервера о том, что тестирование началось), он отсылает серверу запрос C-READY. Если с сервером все в порядке, то ответы, в зависимости от ситуации, могут быть следующие:

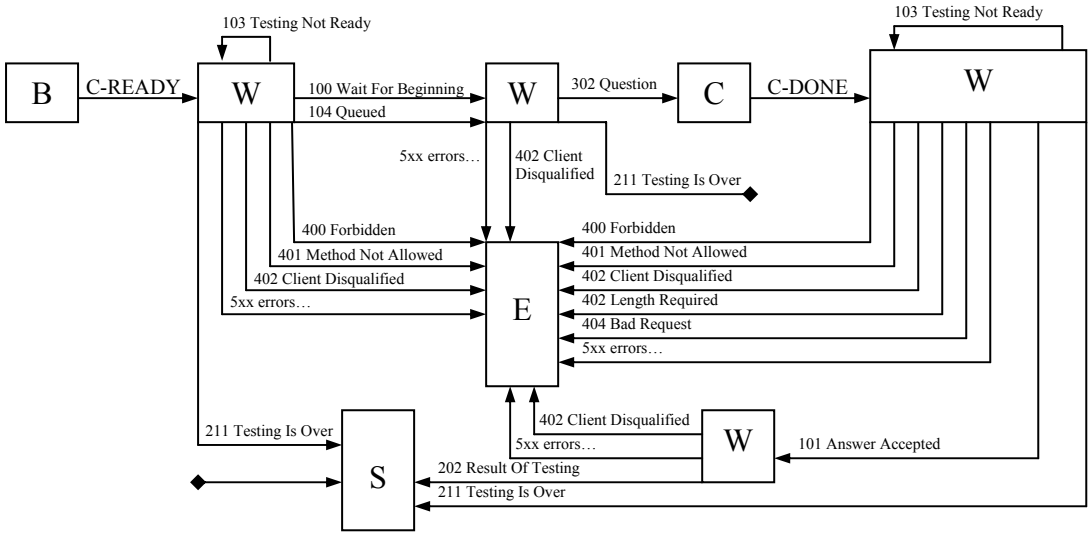
- если тестирование имеет статус «активное», сервер отсылает клиенту уведомление о получении запроса – 104 (*Queued*). После получения от М-клиента очередного вопроса, сервер отсылает его клиенту с помощью ответа 302 (*Question*). Если М-клиент возвращает сообщение о том, что данный клиент тестирование прошел, то сервер отвечает 211 (*Testing Is Over*);
- если тестирование имеет статус «готово», то есть оно еще не началось, сервер посылает ответ 100 (*Wait For Beginning*);
- если тестирование имеет статус «не готово» - ответ 103 (*Testing Not Ready*);
- если тестирование завершено – ответ 211 (*Testing Is Over*).

Клиент, получив ответ 302, отображает вопрос пользователю. Когда пользователь отвечает на вопрос и инициирует отправку решения, клиент посылает запрос C-DONE, содержащий в себе ответ на вопрос. Поведение сервера в данном случае опять зависит от текущей ситуации:

- если с помощью потребностей, которые клиент отсылает вместе с ответом на вопрос, сервер не смог определить того тестера, который сможет проверить данное решение, то сервер посылает ответ 404 (*Bad Request*) с описанием ошибки в заголовке Message;
- если подходящий тестер найден, то сервер ставит решение в очередь на проверку и возвращает ответ 101 (*Answer Accepted*); после того, как решение проверено, результат проверки возвращается сервером вместе с ответом 202 (*Result Of Testing*);
- если же тестирование завершено или не готово, то север посылает, соответственно, 211 (*Testing Is Over*) и 103 (*Testing Not Ready*).

Теперь, для того чтобы получить новый вопрос, клиент должен снова отправить серверу запрос C-READY.

Диаграмма состояний:



C-DONE

Назначение: возвращает серверу ответ клиента на заданный вопрос.

Параметры: нет.

Заголовки:

- Requirements: потребности по проверке решения. Строковые идентификаторы, разделенные символом ‘,’. По данным потребностям сервер определяет того тестера, который может проверить данный ответ.
- Content-Length: длина тела ответа.

Допустимые каналы: КУ.

Описание:

Результат ответа на вопрос возвращается клиентом посредством запроса C-DONE. Данный запрос служит только для возврата ответа на вопрос, он не сообщает серверу, что клиент готов принять новый вопрос. Клиент должен инициировать новую команду C-READY–C-DONE, послав запрос C-READY.

Текст ответа идет в теле запроса. Длина тела указана в заголовке Content-Length.

В случае если сервер в ответ на запрос C-READY возвращает некорректные данные (например, клиент по ошибке подключился не к тому процессу тестирования), то клиент может послать запрос LOGOUT с описанием произошедшей ошибки в заголовке Reason.

Диаграмма состояний:

См. диаграмму состояний для запроса C-READY.

M-READY

Назначение: сообщает серверу о готовности М-клиента выполнить очередное действие.

Параметры: нет.

Заголовки: нет.

Допустимые каналы: M.

Описание:

После того, как M-клиент подключился к серверу, открыл M-канал и отдал серверу тестовый пакет для того тестирования, которое сервер поручил вести этому клиенту, M-клиент готов принимать от сервера заявки на выдачу очередного вопроса для одного из клиентов-участников. Для этого он посылает запрос M-READY серверу.

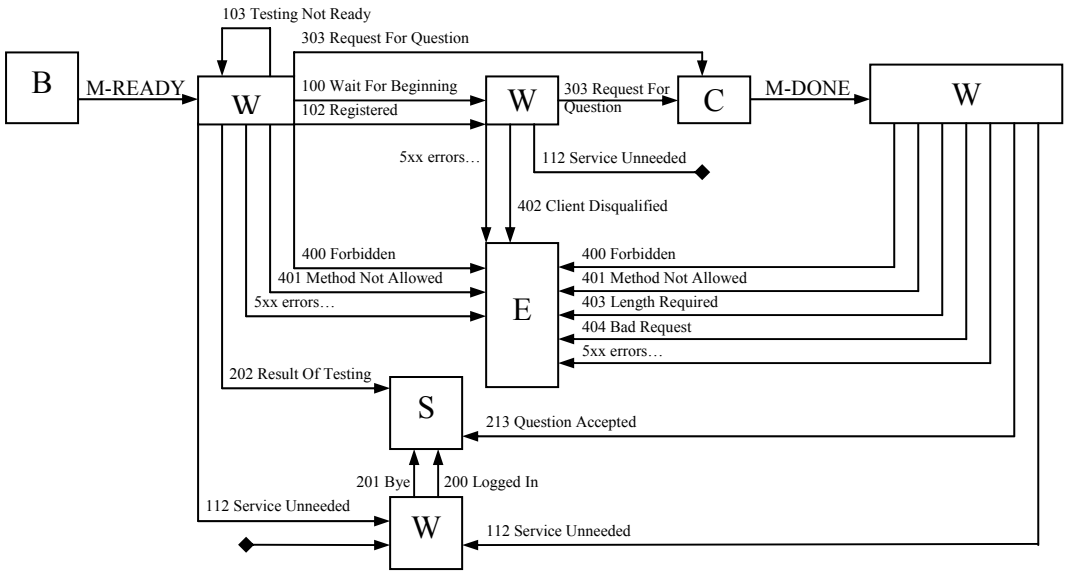
Ответ сервера может быть одним из следующих:

- если тестирование имеет статус «активное» и у сервера есть готовая заявка для M-клиента, то он отсылает эту заявку M-клиенту с ответом 303 (*Request For Question*)
- если тестирование имеет статус «активное» и у сервера есть готовое уведомление о проверке очередного ответа тестером, то сервер отсылает это уведомление M-клиенту с ответом 202 (*Result Of Testing*). На этом команда M-READY завершается и M-клиенту необходимо снова посылать запрос M-READY для того, чтобы получить новую заявку или уведомление.
- если очередь заявок и уведомлений пуста, то сервер посылает уведомление 102 (*Registered*), говорящее о том, что состояние готовности M-клиента отмечено сервером и при первой возможности он отправит M-клиенту необходимую информацию при помощи одного из перечисленных выше ответов;
- если тестирование имеет статус «готово», то есть оно еще не началось, сервер посылает ответ 100 (*Wait For Beginning*);
- если тестирование имеет статус «не готово» - ответ 103 (*Testing Not Ready*);
- если тестирование завершено – ответ 112 (*Service Unneeded*).

Если M-клиент получает ответ 202 (*Result Of Testing*), то на этом выполнение команды M-READY завершается. Ответ сервера 202 (*Result Of Testing*) содержит результат тестирования клиентского ответа, выданный тестером.

Получив ответ 303 (*Request For Question*), M-клиент анализирует его, выбирает очередной вопрос для пользователя или формирует ответ, что данный пользователь ответил на все вопросы. Эту информацию M-клиент посылает серверу при помощи запроса M-DONE. Сервер посылает уведомление о том, что информация от M-клиента получена, посылая ему ответ 213 (*Question Accepted*).

Диаграмма состояний:



M-DONE

Назначение: возвращает серверу очередной вопрос для клиента.

Параметры:

- **finished** – указывает, что клиент прошел все тестирование.

Заголовки:

- Content-Length: длина тела ответа
- Client-Code: кодовое слово клиента

Допустимые каналы: M.

Описание:

После того, как сервер послал M-клиенту заявку на вопрос для клиента, M-клиент формирует необходимое сообщение для данного клиента и посылает его серверу вместе с запросом M-DONE. Данный запрос служит только для передачи серверу вопроса, он не сообщает серверу, что M-клиент готов обрабатывать новые заявки или уведомления. Для того, чтобы показать, что M-клиент снова готов обслуживать тестирование, он должен послать запрос M-READY.

Текст ответа идет в теле запроса. Длина тела указана в заголовке Content-Length.

В случае если сервер в ответ на запрос M-READY возвращает некорректные данные (например, просит вопрос для несуществующего клиента), то M-клиент отвечает командой M-DONE с непустым заголовком Error, в котором он сообщает информацию об ошибке, и пустым телом. После этого сервер внесет сообщение об ошибке в лог и переведет процесс тестирования в состояние «не готово».

M-клиент может послать запрос с параметром finished и пустым телом, что будет означить, что тестирование для клиента, чье кодовое слово содержится в заголовке Client-Code, закончилось.

Диаграмма состояний:

См. диаграмму состояний для запроса M-READY.

TTP

Назначение: возвращает серверу всю необходимую для тестера информацию, такую как, например, тексты тестов, фильтров, информацию о задачах, их количестве и т.д. Форма и содержание тестового пакета определяются конкретным процессом тестирования.

Параметры: нет.

Заголовки:

- TId: идентификатор процесса тестирования, для которого предназначается данный тестовый пакет.
- Content-Length: длина тела запроса.

Допустимые каналы: M.

Описание:

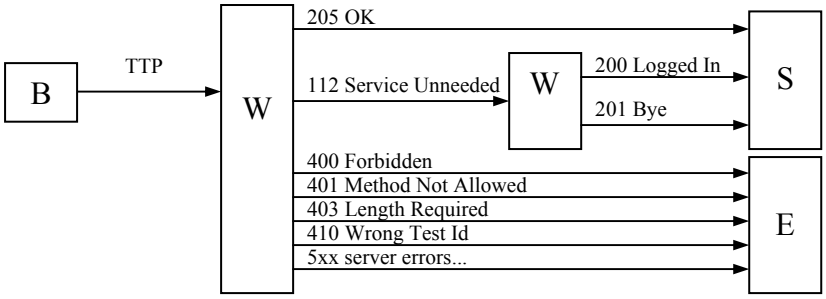
Вместе с ответом, подтверждающим открытие M-канала 200 (*Logged In*), сервер посылает также идентификатор того процесса тестирования, в котором зарегистрирован M-клиент. После этого M-клиент должен передать серверу тестовый пакет для данного тестирования. Формат и содержимое пакета сервером никак не анализируются. Тестовый пакет передается в неизменном виде всем тестерам, запрашивающим его.

Сервер, получив пакет, посылает ответ 205 (*OK*).

Если в момент проведения процесса тестирования, содержимое тестового пакета меняется, M-клиент должен вместо очередного запроса M-READY инициировать запрос TTP. Сервер заменит пакет, находящийся в кэше, на новый и разошлет всем тестерам уведомление, что необходимо поменять тестовый пакет.

Если тестирование уже завершено, то сервер посылает ответ 112 (*Service Unneeded*) и помещает M-клиента в специальный пул свободных компонент, где он ждет в течение некоторого времени появления подходящего процесса тестирования. Если такой процесс тестирования так и не появляется, сервер отключает данную компоненту, посылая ей ответ 201 (*Bye*). Если идентификатор, указанный в заголовке TId, не соответствует тому процессу тестирования, к которому M-клиент подключен, то сервер посылает ответ 410 (*Wrong Test Id*).

Диаграмма состояний:



GTP

Назначение: запрашивает тестовый пакет для процесса тестирования.

Параметры: нет.

Заголовки:

- TId: идентификатор того процесса тестирования, чей тестовый пакет запрошен.

Допустимые каналы: T.

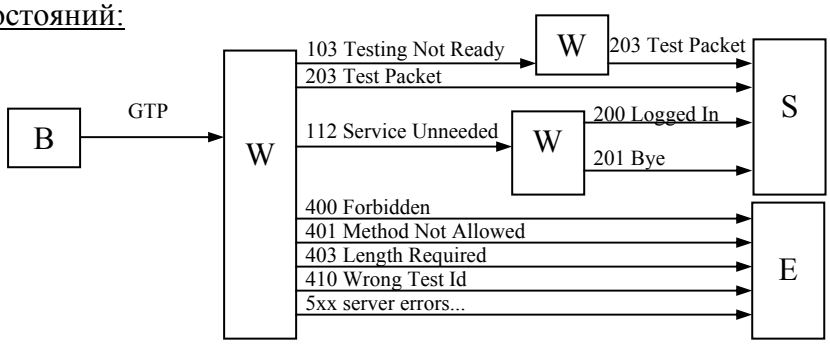
Описание:

После регистрации на сервере, прежде, чем отправить запрос T-READY, говорящий о готовности тестера проверять задачи, тестер должен сформировать тестовую базу, загрузив с сервера тестовый пакет того процесса тестирования, в котором сервер его зарегистрировал.

В случае успешного выполнения сервер возвращает ответ 203 *(Test Packet)* с запрашиваемой информацией. Если нет M-клиента, сервер посылает 103 (*Testing Not Ready*).

Если тестирование уже завершено, то сервер посылает ответ 112 (*Service Unneeded*). Если идентификатор, указанный в заголовке TId, не соответствует тому процессу тестирования, к которому M-клиент подключен, то сервер посылает ответ 410 (*Wrong Test Id*).

Диаграмма состояний:



T-READY

Назначение: сообщает серверу о готовности тестера принять очередную задачу на проверку.

Параметры: нет.

Заголовки: нет.

Допустимые каналы: T.

Описание:

Открытие T-канала еще не позволяет серверу отправлять тестеру задачи на проверку.

Именно для решения этих проблем и служит запрос T-READY, отсылаемый тестером тогда, когда он готов принимать задачи. Запрос T-READY является первым запросом в составной команде T-READY – T-DONE.

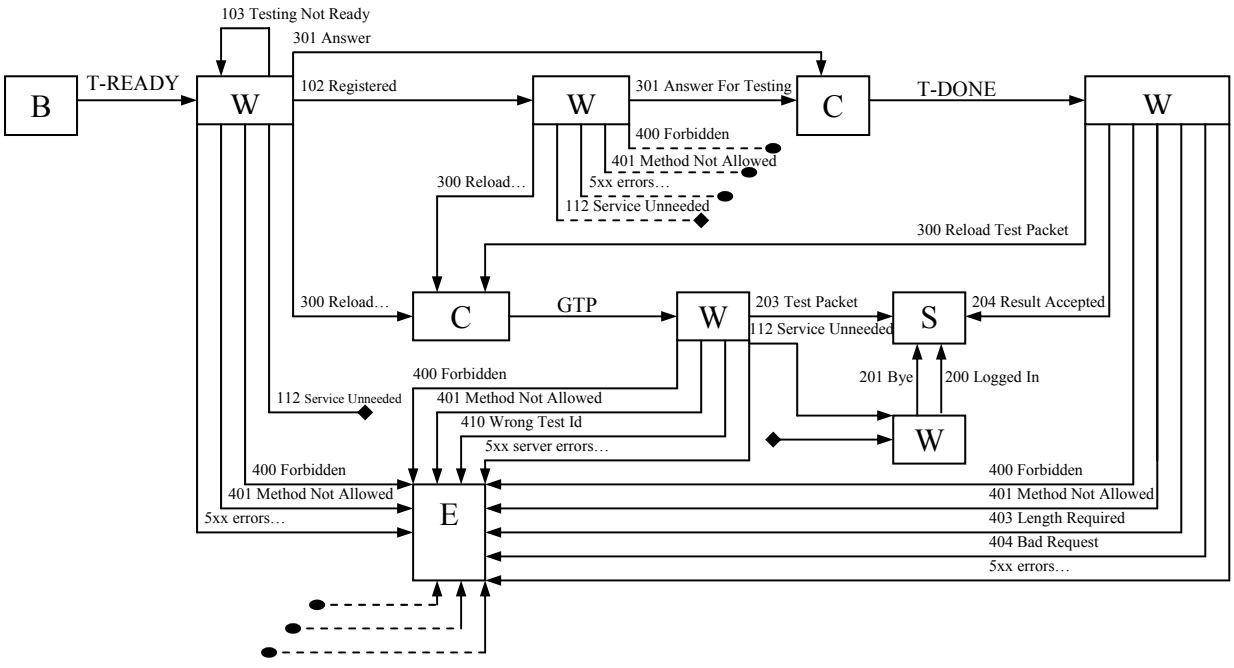
Когда тестер готов принять задачу на проверку, он посылает серверу запрос T-READY. Если с сервером все в порядке, то он либо сразу же отсылает ему клиентский ответ на проверку с помощью ответа 301 *(Answer)*, либо, если такового нет, отмечает тестера как свободного и посылает ему ответ 102 *(Registered)*. Во втором случае, когда появляется клиентский ответ и сервер решает отправить его на проверку, он делает это также с помощью ответа 301 *(Answer)* и вдобавок к этому помечает тестера как занятого, таким образом, предотвращая отсылку одновременно двух задач на одну тестирующую машину.

Тестер, получив ответ 301 *(Answer)*, проверяет клиентский ответ и возвращает результат проверки серверу с помощью запроса T-DONE. Сервер (опять же – если в его работе не произошло никаких фатальных ошибок) отвечает на этот запрос сообщением 204 *(Result Accepted)*, и на этом общение сервера и тестера заканчивается, так как составная команда T-READY – T-DONE выполнена полностью. Теперь, для того чтобы получить новый клиентский ответ на проверку, тестер должен снова отправить серверу запрос T-READY.

Существует лимит ожидания сервером запроса от тестера T-DONE, по истечению которого сервер закрывает соединение с тестером и переадресует задачу другому тестеру.

Вместо ответов 102 (*Registered*), 301 (*Answer*) и 204 (*Result Accepted*) сервер может послать либо ответ 300 *(Reload Test Packet)* – в этом случае тестер должен, перед тем, как послать запрос T-READY, заново запросить тестовый пакет у сервера посредством запроса GTP; либо ответ 112 (*Service Unneeded*).

Диаграмма состояний:



T-DONE

Назначение: возвращает результат тестирования задачи серверу.

Параметры: нет.

Заголовки: нет.

Допустимые каналы: T.

Описание:

Результат проверки задачи возвращается тестером посредством запроса T-DONE. Данный запрос служит только для возврата результата проверки – он не сообщает серверу, что тестер свободен и готов принять новую задачу на проверку. Тестер должен инициировать новую команду T-READY – T-DONE, послав запрос T-READY.

Если все в порядке, сервер отвечает 204 (*Result Accepted*). Вместо ответа 204 (*Result Accepted*), сервер может послать ответ 300 (*Reload Test Packet*), заставив тестера заново загрузить пакет тестов. Если тестирование уже завершено, то сервер посылает ответ 112 (*Service Unneeded*).

Диаграмма состояний:

См. диаграмму состояний для запроса T-READY.

INIT

Назначение: заставляет сервер перезагрузить все свои динамические настройки из базы данных.

Параметры: нет.

Заголовки:

- TId: перечисленные через запятую идентификаторы тех процессов тестирования, чьи настройки необходимо загрузить. Если заголовок не указан, то обновляются все процессы тестирования.

Допустимые каналы: A.

Описание:

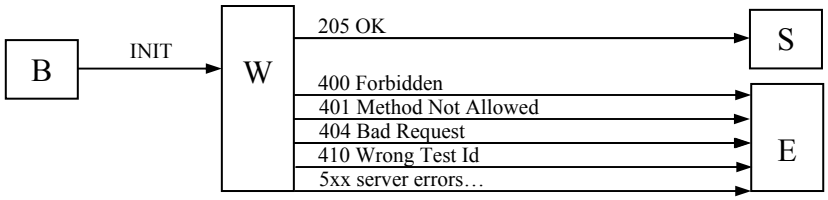
Все настройки сервера можно разделить на две группы: динамические – те, которые загружаются из базы данных, и статические – которые загружаются из конфигурационного файла в момент запуска сервера. Помимо динамических настроек, сервер также обладает динамическим статусом, который отображается им в режиме реального времени в базу данных. К статусу относятся:

- состояния всех загруженных сервером процессов тестирования;
- подключенные к системе клиентские компоненты: тестеры, M-клиенты, клиенты, рейтинг-серверы, административные клиенты.

Динамические настройки и статус сервера вместе составляют динамическую конфигурацию сервера. Эта конфигурация полностью содержится в базе данных. В ход работы сервера можно вмешиваться, изменяя данную динамическую конфигурацию и посылая серверу запрос INIT. Получив, запрос INIT, сервер считывает из базы данных и применяет как свои динамические настройки, так и значение статуса, которое возможно было вручную изменено администратором сервера (например, администратор может вручную изменить распределение тестеров по различным однотипным процессам тестирования после того, как сервер выполнил это распределение самостоятельно).

В случае успеха сервер возвращает ответ 205 *(OK)*. Если произошла какая-либо ошибка (например, некорректные данные в базе данных), то сервер возвращает ошибку 500 (*Internal Server Error*) с описанием ошибки в заголовке Message. Если ни одни идентификатор в заголовке TId не соответствует реальным процессам тестирования, то сервер возвращает 410 (*Wrong Test Id)*. Если хотя бы один идентификатор соответствует реальному процессу тестирования, то все те идентификаторы, которые не соответствуют, просто игнорируются.

Диаграмма состояний:



LOG

Назначение: запрашивает обобщенный лог указанного процесса тестирования.

Параметры:

- **with-last-timestamp**: если указать, то сервер помимо таблицы всего обобщенного лога вернет и время добавления последней записи.

Заголовки:

- TId: идентификатор процесса тестирования, чей лог запрашивается.

Допустимые каналы: P.

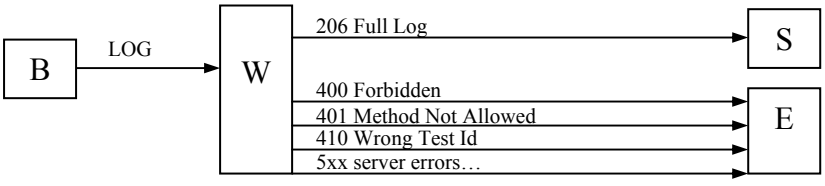
Описание:

Запрашивает обобщенный лог для указанного в заголовке процесса тестирования. Клиент не должен рассчитывать на то, что сервер вернет таблицу с уже отсортированными записями.

В случае успеха сервер возвращает ответ 206 *(Full Log)* с запрашиваемой информацией. Если TId не соответствует номеру ни одного процесса тестирования, имеющегося в базе данных, то сервер возвращает ответ 410 (*Wrong Test Id*).

Информация о логе возвращается сервером в виде xml-дерева, структура которого приводится в описании соответствующего ответа сервера.

Диаграмма состояний:



LOG-PART

Назначение: запрашивает последние записи из таблицы обобщенного лога.

Параметры: нет.

Заголовки:

- TId: идентификатор процесса тестирования, чей лог запрашивается.

- From: запрашивает все записи из таблицы обобщенного лога, которые были добавлены в таблицу после времени, указанного в данном заголовке.

Допустимые каналы: P.

Описание:

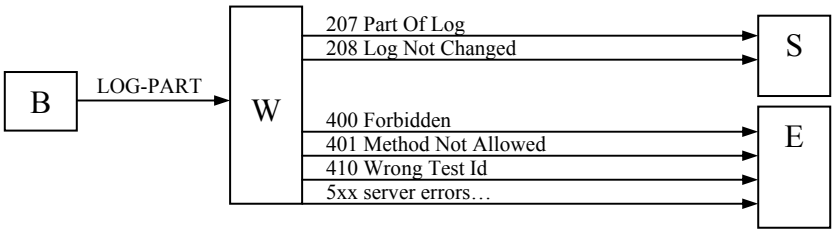
Для вывода лога клиент может использовать следующий механизм. Сначала он запрашивает всю таблицу лога какого-либо процесса тестирования, используя запрос LOG с параметром with-last-id. После этого он через фиксированные промежутки времени посылает запрос LOG-PART с имеющимся идентификатором. Запрос LOG-PART помимо данных возвращает также новое значение идентификатора, которое клиент пошлет в следующий раз.

На запрос LOG-PART сервер возвращает не таблицу с логом, а последние его записи. Если значение, указанное в заголовке From, равно идентификатору последней записи в базе данных (то есть с момента последнего запроса никаких новых данных в базу не было добавлено), то сервер возвращает ответ 208 *(Log Not Changed)*.

В случае успеха сервер возвращает ответ 207 *(Part Of Log)* с запрашиваемой информацией. Если TId не соответствует номеру ни одного процесса тестирования, имеющегося в базе данных, то сервер возвращает ответ 410 (*Wrong Test Id*).

Информация о логе возвращается сервером в виде xml-дерева, структура которого приводится в описании соответствующего ответа сервера.

Диаграмма состояний:



PROFILE

Назначение: запрашивает профиль процесса тестирования.

Параметры: нет.

Заголовки:

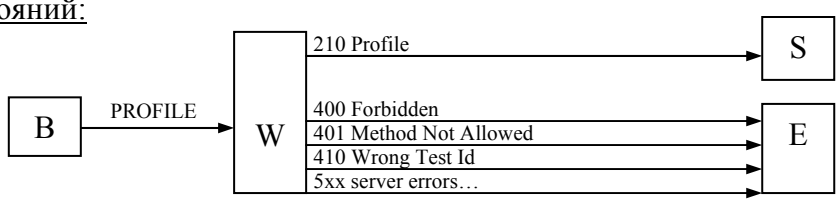
- TId: идентификатор процесса тестирования, профиль которого запрашивается.

Допустимые каналы: P.

Описание:

Профиль процесса тестирования – это информация, характеризующая процесс тестирования. Данная информация хранится в базе данных. В ее формировании сервер не принимает никакого участия. При запросе PROFILE сервер берет ее из базы данных и передает клиенту вместе с ответом 210 *(Profile)*. Если TId не соответствует номеру ни одного процесса тестирования, имеющегося в базе данных, то сервер возвращает ответ 410 (*Wrong Test Id*).

Диаграмма состояний:



10. Ответы

10.1. Синтаксис ответа

Ответ имеет следующий синтаксис в БНФ-нотации:

ответ := <ответ> <CRLF> [заголовки] <CRLF> [тело]
<ответ> := <протокол>/<версия> <SP> <код> <SP> <текст ответа>
<код> := 1xx | 2xx | 3xx | 4xx | 5xx
1xx := 100 | 101 | 102 | 103 | 104
2xx := 200 | 201 | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 | 210 | 211 | 212 | 213 | 220
3xx := 300 | 301 | 302 | 303
4xx := 400 | 401 | 402 | 403 | 404 | 410
5xx := 500 | 501
<текст ответа> := {для каждого кода свой}
<протокол> := OLYMPIA
<версия> := [0-9]+ . [0-9]+
заголовки := заголовок <CRLF> | заголовок <CRLF> заголовки
заголовок := <Имя заголовка>: <значение>
<Имя заголовка> := TId | Message | From | Content-Length | Client-Code | Timestamp
тело := {текст, предусмотренный ответом, длина которого указана в Content-Length}
<SP> := _ | \t
<CRLF> := \n

Длина строки команды и заголовков не может превосходить 1024 символов. Количество заголовков также не может превосходить 1024. Если ответ подразумевает наличие тела, то среди заголовков обязательно должен присутствовать заголовок Content-Length с длиной тела в байтах.

Регистр команд, параметров, заголовков, а также названия протокола не важен. Концом заголовка ответа (включающего строку команды и опциональный блок заголовков) является пустая строка. <код> представляет собой последовательность из цифр. Порядок заголовков не имеет значения, имя заголовка отделяется от его значения двоеточием. Заголовок, а также строка команды не должны содержать начальных пробелов.

10.2. Описание ответов

1) *100 Wait For Beginning*
   Назначение: посылается сервером клиенту при установлении КУ- и Р-каналов, когда тестирование еще не началось.

2) *101 Answer Accepted*
   Назначение: возвращается клиенту на запрос C-DONE. Указывает на то, что ответ успешно поставлен в очередь на проверку.

3) *102 Registered*
   Назначение: возвращается тестеру и М-клиенту на запрос *-READY. Указывает на то, что в настоящий момент никаких заданий в очереди нет и соответствующая клиентская компонента помечена как свободная.

4) *103 Testing Not Ready*
   Назначение: возвращается на запрос любой компоненте тогда, когда необходимый процесс тестирования имеет состояние «не готов». При этом все последующие запросы клиента игнорируются. Когда процесс тестирования возобновляется, сервер посылает клиенту тот ответ, который предусматривается протоколом, используя при этом данные, посланные пользователем при первом запросе.

5) *104 Queued*
Назначение: возвращается клиенту на запрос C-READY. Указывает на то, что заявка на очередной вопрос для клиента поставлена в очередь соответствующему М-клиенту.

6) *112 Service Unneeded*
Назначение: возвращается сервером в случае, когда в услугах подключаемого М-клиента или тестера еще или уже нет необходимости. Соответствующая клиентская компонента переводится в пул свободных компонент, где ожидает времени, когда появится подходящий процесс тестирования. Если по прошествии определенного интервала времени подходящий процессов тестирований не появляется, сервер автоматически посылает ответ 201 (*Bye*) и закрывает канал связи. Если появляется процесс тестирования, который может быть обслужен данной компонентой, то сервер регистрирует ее в этом процессе тестирования и посылает ей ответ 200 (*Logged In*) с идентификатором этого процесса тестирования.

7) *200 Logged In*
Назначение: сообщает об успешном открытии виртуального канала передачи данных.

Заголовки:
- TId: возвращается тестеру и М-клиенту для указания того, в каком процессе тестирования они зарегистрированы.

8) *201 Bye*
Назначение: сообщает об успешном закрытии виртуального канала передачи данных.

Заголовки:
- Message: указывается причина закрытия канала, если таковой является ошибка.

9) *202 Result Of Testing*
Назначение: возвращает клиенту результат тестирования программы. В теле сообщения идет Content-Length байт, которые вернул тестер при проверке данной задачи.

Заголовки:
- Content-Length: длина тела в байтах.
- Timestamp: время поступления в систему ответа на вопрос.

10) *203 Test Packet*
Назначение: возвращает тестеру тестовый пакет процесса тестирования.

Заголовки:
- TId: идентификатор процесса тестирования, тестовый пакет которого содержится в теле.
- Content-Length: длина тела ответа в байтах.

11) *204 Result Accepted*
Назначение: возвращается тестеру в ответ на запрос T-DONE. Сообщает о том, что результат проверки ответа принят.

12) *205 OK*

Назначение: успешное выполнение команды.

Заголовки:

- Message: текстовое сообщение о той команде, которая была выполнена.

13) *206 Full Log*

Назначение: возвращает всю таблицу обобщенного лога.

Заголовки:

- TId: идентификатор соответствующего процесса тестирования.
- Content-Length: длина тела ответа в байтах.

Описание:

Тело ответа содержит обобщенный лог, представленный в XML-формате.

```
<xml version="1.0" [compression="zip"]/>
<log TId="test_id">
    <item id="item_id">
            <from type="from_type" name="from_name"/>
            <to type="to_type" name="to_name"/>
            <dateEntered>
                    <year>год поступления</year>
                    <month>месяц поступления</month>
                    <day>день поступления</day>
                    <hour>час поступления</hour>
                    <minute>минута поступления</minute>
                    <second>секунда поступления</second>
            </dateEntered>
            <dateLeft>
                    <year>год ухода</year>
                    <month>месяц ухода</month>
                    <day>день ухода</day>
                    <hour>час ухода</hour>
                    <minute>минута ухода</minute>
                    <second>секунда ухода</second>
            </dateLeft>
            <body compression="compression_method">тело_сообщения</body>
    </item>
</log>
```

1. id - идентификатор записи (уникален для всех записей в базе данных сервера)
2. TId - идентификатор процесса тестирования
3. *from_type* – тип источника (meta, tester)
4. *from_name* – имя источника (GUID для тестера, пусто для M-клиента)
5. *to_type* – тип приемника (meta, tester)
6. *to_name* – имя приемника (GUID для тестера, пусто для M-клиента)
7. *compression_method* – метод кодирования данных узла. Может быть либо "BASE64", либо "ZIP+BASE64". Тело сообщения должно быть обязательно кодировано методом, на выходе которого получается безопасная с точки зрения xml последовательность символов. В текущей реализации сервера подобным методом кодирования является алгоритм BASE64. Предварительно текст может быть сжат алгоритмом Лемпеля-Зива.

Тег xml идет в самом начале документа и может опционально содержать атрибут compression, принимающий только одно значение ‘zip’. В этом случае все, что идет сразу за этим тегом (после символа переноса строки), упаковано соответствующим методом.

Лог, выдаваемый сервером в ответ на запрос LOG или LOG-PART, содержит только сообщения M-клиента (тип источника meta), являющиеся вопросами для клиента, и сообщения тестера (тип источника tester), являющиеся результатом проверки ответа клиента на вопрос. Сами ответы в логе не передаются.

14) *207 Part Of Log*

Назначение: возвращает последние записи из таблицы лога.

Заголовки:

- From: время последней записи в таблице лога
- TId: идентификатор соответствующего процесса тестирования
- Content-Length: длина тела ответа в байтах

Описание:

Тело ответа содержит информацию о логе, представленную в XML-формате. См. ответ 206 (*Full Log*).

15) *208 Log Not Changed*

Назначение: возвращает сообщение о том, что с момента последнего запроса лог не изменялся.

Заголовки:

- TId: идентификатор соответствующего процесса тестирования
- From: содержит то же время, что и соответствующий рейтинговый запрос

16) *209 Testing Started*

Назначение: отсылается сервером соответствующим клиентским компонентам в момент начала проведения процесса тестирования.

17) *210 Profile*

Назначение: возвращает профиль процесса тестирования

Заголовки:

- TId: идентификатор соответствующего процесса тестирования
- Content-Length: длина тела ответа в байтах

18) *211 Testing Is Over*

Назначение: возвращается сервером, когда процесс тестирования завершается или уже завершен.

19) *213 Question Accepted*

Назначение: возвращается M-клиенту в ответ на запрос M-DONE. Сообщает о том, что вопрос для клиента принят.

20) *220 <имя сервера> at <имя хоста>*

Назначение: строка приветствия сервера. Возвращается любому клиенту в момент установления сетевого соединения.

21) *300 Reload Test Packet*

Назначение: Сообщает тестеру о том, что необходимо заново загрузить тестовый пакет

Заголовки:

- TId: идентификатор соответствующего процесса тестирования

22) *301 Answer*

Назначение: Возвращает тестеру ответ для тестирования.

Заголовки:

- Content-Length: длина тела ответа в байтах.

Описание:

В теле ответа идет сам клиентский ответ.

23) *302 Question*

Назначение: содержит вопрос для клиента-участника тестирования.

Заголовки:

- Content-Length: длина тела ответа в байтах.

Описание:

В теле ответа идет сам вопрос для клиента.

24) *303 Request For Question*

Назначение: Возвращает М-клиенту заявку на очередной вопрос для клиента

Заголовки:

- Client-Code: кодовое слово клиента
- Timestamp: время поступления в систему заявки.

25) *400 Forbidden*

Назначение: сообщает о том, что запрос не может быть выполнен из-за отсутствия соответствующих прав.

Заголовки:

- Message: сообщение об ошибке.

26) *401 Method Not Allowed*

Назначение: возвращается клиенту при попытке послать защищенный запрос, не соответствующий каналу связи.

27) *402 Client Disqualified*

Назначение: посылается клиенту в ответ на запросы, если данный клиент дисквалифицирован.

28) *403 Length Required*

Назначение: возвращается клиенту в ответ на запрос, содержащий тело, если в данном запросе опущен заголовок Content-Length.

29) *404 Bad Request*

Назначение: возвращается в ответ на ошибочный запрос. Ошибочным считается запрос, у которого либо синтаксис команды некорректен, либо в семантике содержатся ошибки.

Заголовки:

- Message: текстовое описание ошибки.

30) *410 Wrong Test Id*

Назначение: возвращается сервером в случае указания клиентом неверного идентификатора процесса тестирования.

31) *500 Internal Server Error*

Назначение: возвращается при внутренней ошибке сервера.

Заголовки:

- Message: текстовое описание ошибки.

32) *501 OLYMPIA Version Not Supported*

Назначение: возвращается клиенту, если в запросе указан протокол, не поддерживаемый сервером.