Описание обмена сообщениями в процессе создания игры и приглашения игроков.

Игрок, который хочет создать новую игру, через приложение агент отправляет сообщение (0.1) на сервер аутентификации со списком аккаунтов, которым разрешен доступ в игру. В ответ сервер аутентификации формирует сообщение, в котором содержится строковое представление идентификатора игры – idGame. Получив идентификатор, создатель игры через приложение агент на информационный сервер отправляет сообщение (0.2), в котором содержится полученный ранее идентификатор игры и список аккаунтов, которым разрешено участие в игре. Информационный сервер, формирует команду (0.3) игровому серверу создать игру с идентификатором и списком участников, переданным в команде. Игровой сервер создает игру с указанным списком аккаунтов и с указанным идентификатором. Агенты пользователей, периодически опрашивая информационный сервер (сообщение 0.4), получают в ответном сообщении идентификатор игры, в которую им предоставлен доступ создателем. Получив идентификатор игры, агент отправляет сообщение серверу аутентификации (0.5), в котором содержится идентификатор игры и параметры его аккаунта. Сервер аутентификации проверяет пользователя по полученному идентификатору игры и выдает ему токен для участия в игре, если этот пользователь был в числе участников. После получения токена, приложение агент формирует поле для игры и отправляет на игровой сервер сообщение (0.6) с идентификатором игры и токеном, полученным от сервера аутентификации. Игровой сервер, получив сообщение (0,6), проверяет токен на валидность и отправляет в ответ приложению агенту список параметров игровых объектов. Агент, получив список, формирует поле с игровыми объектами, согласно полученным параметрам и дает пользователю выбрать объект, которым он будет управлять. После выбора игрового объекта, агент отправляет на игровой сервер запрос (0.7) на управление игровым объектом. Игровой сервер, получив запрос и проверив права пользователя, отправляет положительный ответ в случае достаточных прав пользователя или отрицательный ответ в случае недостаточности прав. В последнем случае процедура выбора игрового объекта повторяется до положительного ответа игрового сервера. Далее агент ожидает начала игры, периодически отправляя на игровой сервер запрос (0.8) о начале игры. Агент, получив время до начала игры, запускает обратный таймер и ожидает начала игры. После старта игры игрок может взаимодействовать со своим игровым объектом, а агент периодически отправляет на сервер сообщение (1) с положением игрового объекта на своем поле, получая в ответ параметры остальных игровых объектов. Также агент периодически отправляет асинхронное сообщение (2) игровому серверу для получения текущих параметров игровых объектов на игровом поле.

Игровой сервер периодически отправляет на сервер статистики сообщение (1.1), в котором указывает идентификатор игры, участников игры и параметры игровых объектов. Сервер статистики из полученных данных ведет статистику игры.

Игрок может во время игры или в любое другое послать сообщение (2.1) посредство агента на сервер статистики, указав идентификатор игры. В ответ сервер статистики отправляет статистику по запрошенной игре.

Узкие места и потенциальные проблемы масштабирования приложения и способы их решения

Наиболее высокие требования к масштабированию будут предъявляться к игровому серверу. Рост числа пользователей и усложнение игровых алгоритмов неизбежно приведет к исчерпанию вычислительных ресурсов игрового сервера. Эта проблема может быть решена увеличением вычислительной способности сервера, однако такое решение может быть экономически нецелесообразным. Увеличение вычислительной способности сервера можно решить путем построения распределенной вычислительной системы, состоящей из микросервисов. При таком подходе возможно два пути решения задачи

1. Разработка микросервисов без изменения ендпоинтов и интерфейсов.
2. Разработка микросервисов и изменением ендпоинтов и интерфейсов.

Первый подход исключает внесение изменений в приложения-агенты, что предпочтительнее. Второй подход предусматривает внесение изменений в приложения-агенты. При таком подходе сообщение от агента будет привязано к определенному ендпоинту. А агенты должны будут знать на какой ендпоинт оправить сообщение. То есть агент должен иметь таблицу соответствия «тип сообщения – эндпоинт сервера».

Еще один способ решить проблему исчерпания вычислительных ресурсов сервера заключается в делегировании части вычислительных задач клиенту. Таким образом, потребуется внесение изменений не только в код игрового сервера, но и в код агентов. Однако и такой подход также не лишен недостатков, главными из которых являются строгие требования к пропускной способности сети и времени отклика.

При росте числа пользователей и их рейтингов, возможно, понадобиться распределять роли и давать привилегии определённым пользователям. Например, роль командира и возможность управления несколькими игровыми объектами, в том числе объектами других пользователей. Роли могут распределяться динамически во время игры. В настоящий момент в архитектуре не предусмотрен механизм распределения ролей и привилегий. Для решения такой задачи может быть разработан микросервис распределения ролей, который динамически на основе статистики игры и рейтинга пользователей будет наделять определенных пользователей особыми правами и наоборот ограничивать их в правах, например за нарушение правил. Сервис ролей будет взаимодействовать с игровым сервером, обновляя роли игроков в списке аккаунтов.

