Спецификация взаимодействия клиента и сервера.

1) Типы запросов клиента и их обработка
2) Формат сообщений, посылаемых с клиента на сервер
3) Формат ответа сервера

Типы запросов клиента к серверу:

- Функциональность клиента состоит в том, чтобы посылать запрос, принимать ответ и его выводить. Никакой обработки информации не нужно.
- Типы сообщений заданы в файлахGameEvent.as и GameEvent.h. ОБЯЗАТЕЛЬНО нужно соблюдать идентичность формата. Если добавились новые события в одном файле, нужно добавить ее в другом файле и добавить в спецификацию.
- Нужно учитывать, что везде может быть FAIL({Reason : string})
- Описание запросов:

Код	Команда	От кого	аргументы	Описание	Возможные ответы
0	Register	Клиент	Id	игрок устанавливает игру	• Success • PlayerAlreadyRegistered
1	Auth	Клиент	Id, password	Авторизация	• Success • PlayerAlreadyAuthorized
2	GetOnlinePlayers	клиент	{}	Получает список онлайн игроков	• Массив Objectов с полями id
3	InviteToPlay	Клиент	Id	Вызывает игрока с id на матч	• Success
4	ReceiveInvite	Сервер	Id	Inform player about challenge with id-player	
5	AcceptInvite	Клиент	Id	Запрос на начало матча с игроком id	• GameStart
6	Move	Клиент	From, to: string	Запрос на ход	• GameState
7	InformMove	Сервер	From, to: string	Оповещает игроков о ходе	
8	GameStart	Сервер	WhiteTurn, White[], black[], fog[]	Оповещает игроков о начале матча	
9	GameEnd	Сервер	MatchResult	Оповещает игроков о конце матча, обновляет рейтинг	
10	AddOnlinePlayer	Сервер	Id	Оповещает ВСЕХ игроков о появлении человека в онлайне	
11	RemoveOnlinePlayer	Сервер	Id	Оповещает ВСЕХ игроков о оффлайн	

				человеке	
12	ViewPlayerInfo	Клиент	Id	Получает информацию об игроке	{gold, experience}
13	GameEvent_MAX				

Формат сообщений, посылаемых с клиента на сервер.[НИЗКОУРОВНЕВЫЙ]

Формат запросов будет выглядеть примерно так: [messageLength, messageCommand, {arguments}], где

- messageLength – ожидаемая длина сообщения в байтах. ~~Жизнь настолько сурова,~~ TCP/IP так устроен, что сообщения необязательно придут сразу же одним сообщением, поэтому обрабатывать запрос нужно только в том случае, когда у нас есть все данные.
- messageCommand – тип запрос клиента, см. Типы запросов
- arguments – аргументы запроса messageCommand. Должен приходить в виде объекта. Пример: {id:3371777, password:”PASSWORD”}. **Внимание!** Нужно строго придерживаться формата названий параметров!

Формат ответа сервера клиенту.[НИЗКОУРОВНЕВЫЙ]

Ответы сервера могут быть разными(начиная от числа, заканчивая массивами фигур), поэтому предлагаю следующий формат ответа:
[messageLength, messageCommand, {answer}], где

- messageLength – длина ответа в байтах.
- respond - запрос | ответ на запрос. //const
- resultCode
- arguments – ответ. Приходит в виде массива данных.