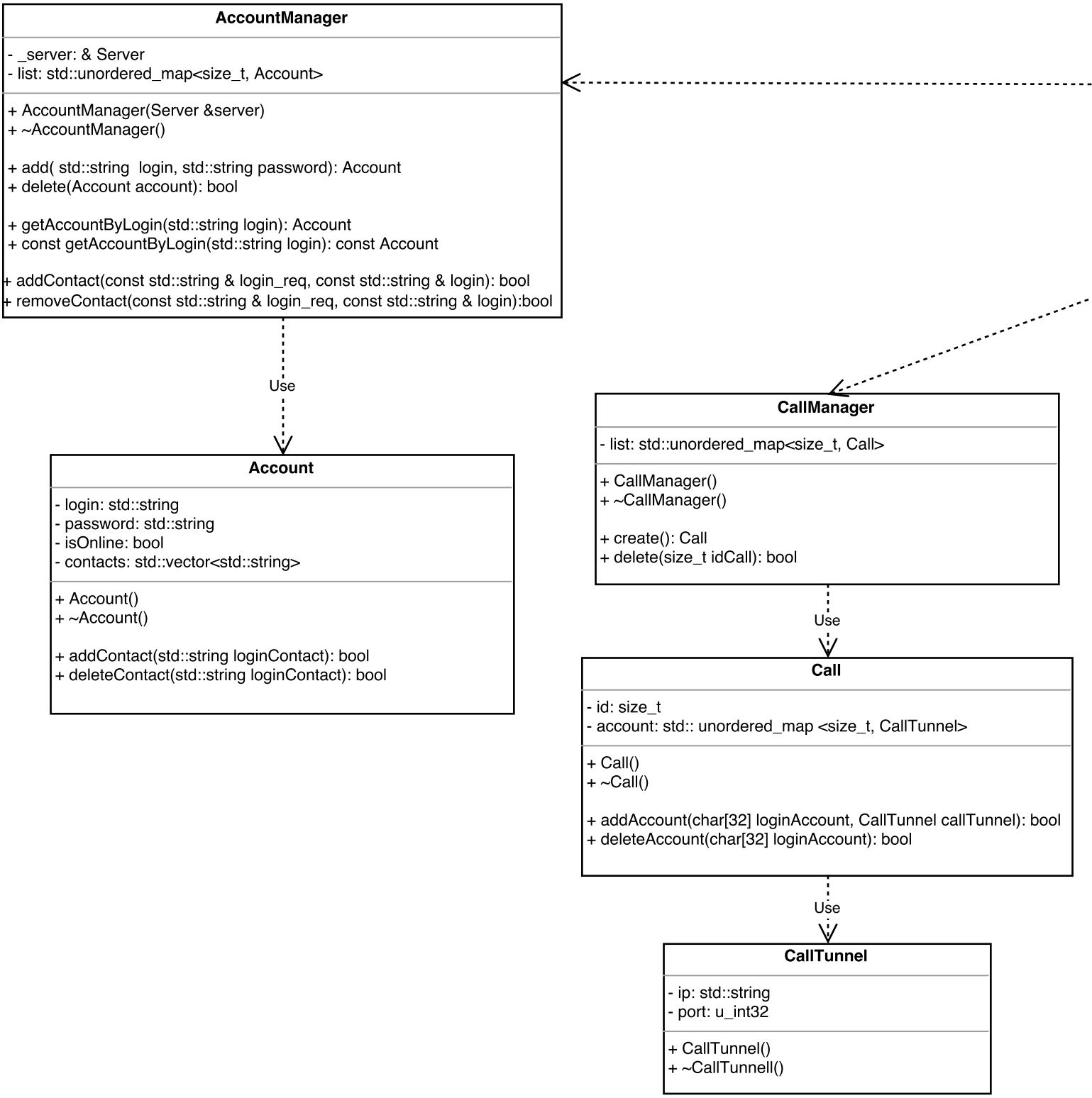
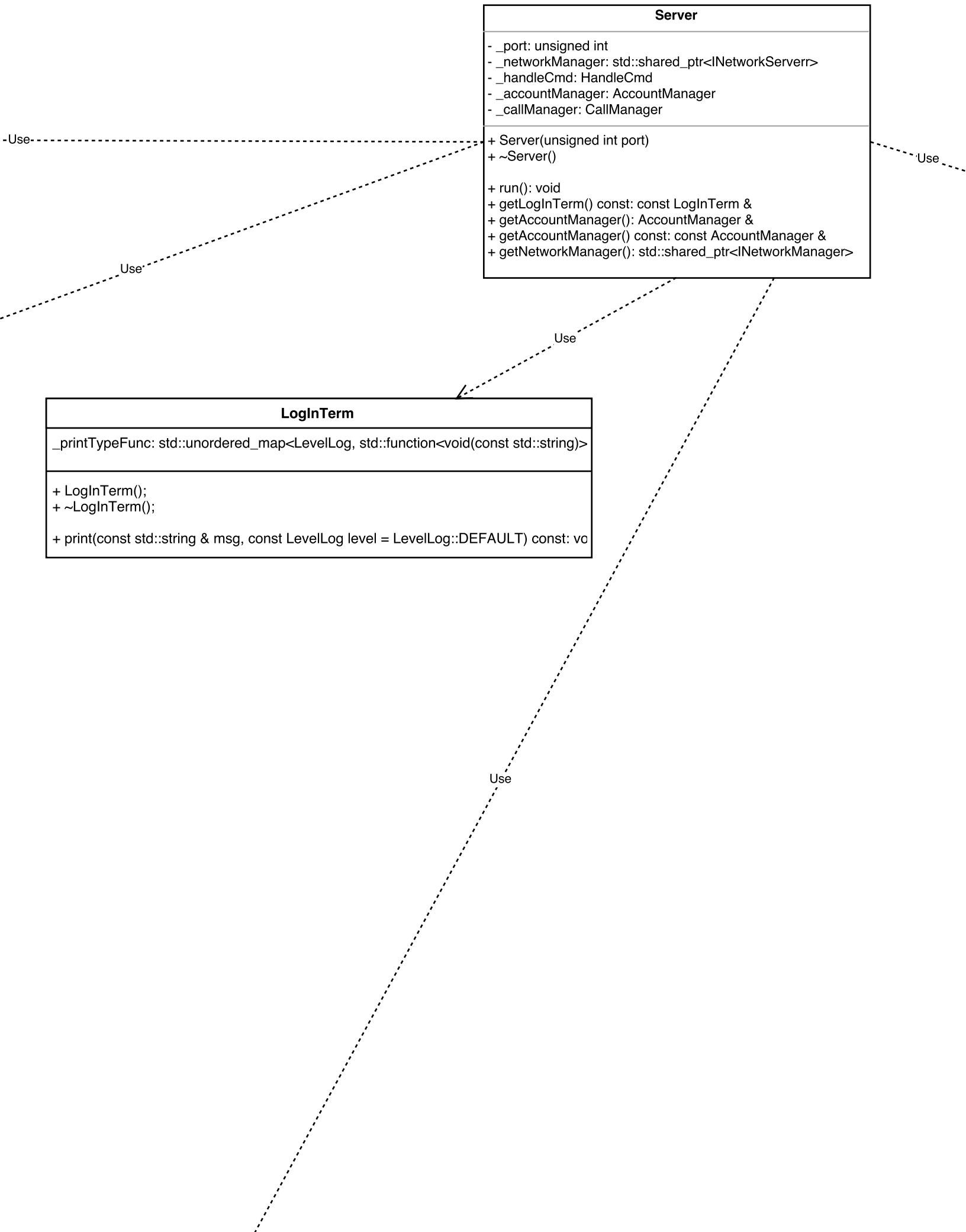
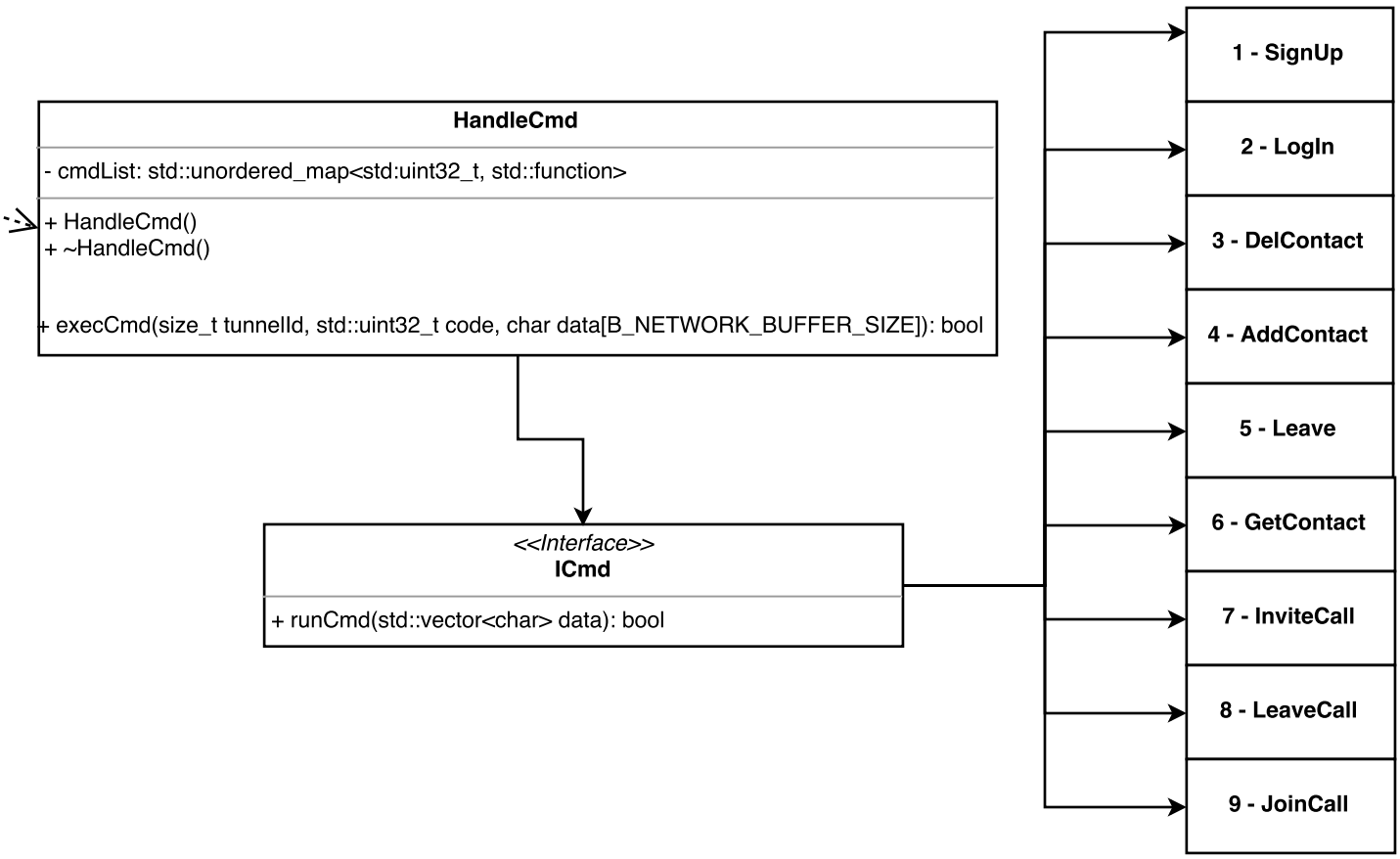
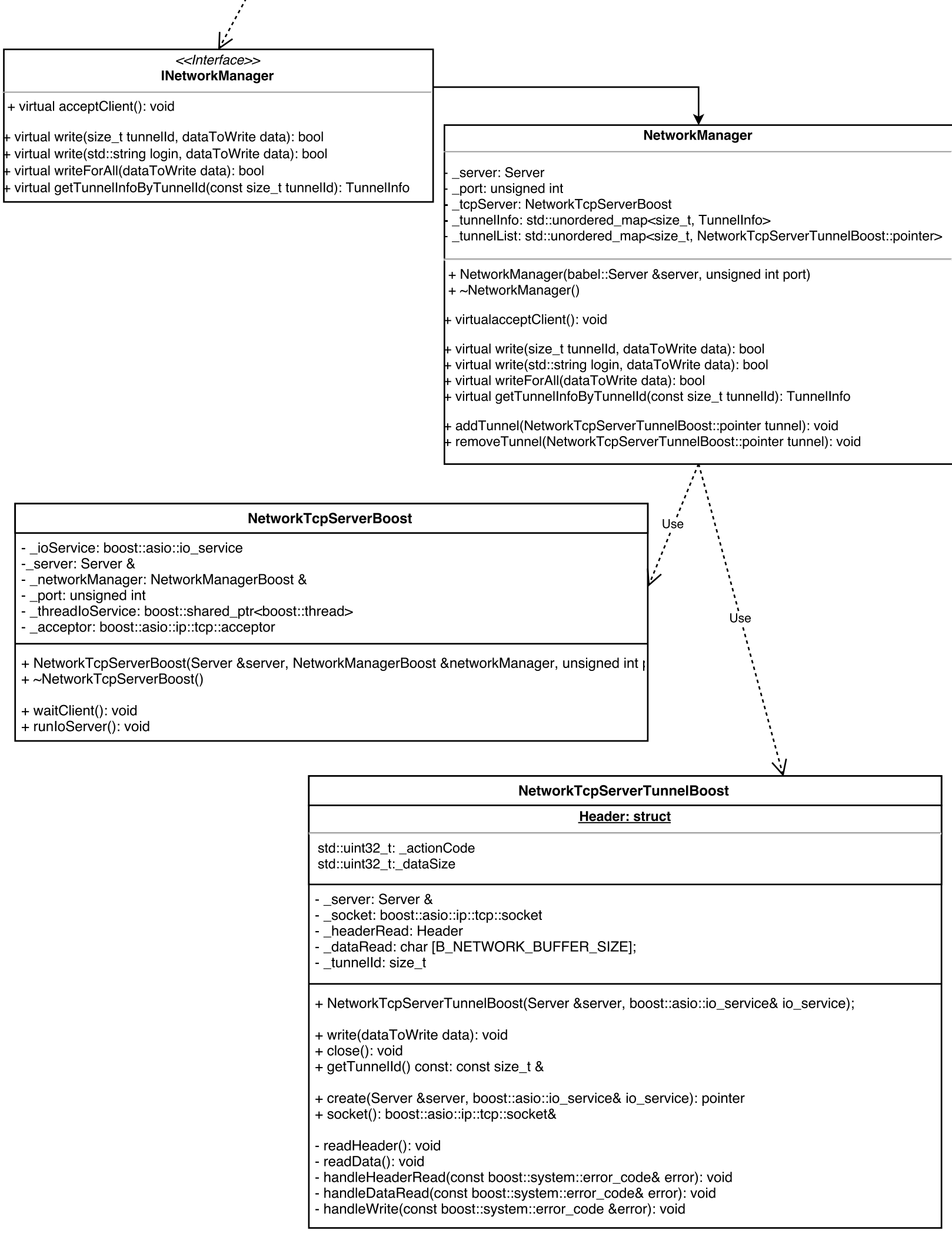

AccountManager
- _server: & Server
- list: std::unordered_map<size_t, Account>
+ AccountManager(Server &server)
+ ~AccountManager()
+ add(std::string login, std::string password): Account
+ delete(Account account): bool
+ getAccountByLogin(std::string login): Account
+ const getAccountByLogin(std::string login): const Account
+ addContact(const std::string & login_req, const std::string & login): bool
+ removeContact(const std::string & login_req, const std::string & login):bool
Use
Account
- login: std::string
- password: std::string
- isOnline: bool
- contacts: std::vector<std::string>
+ Account()
+ ~Account()
+ addContact(std::string loginContact): bool
+ deleteContact(std::string loginContact): bool
CallManager
- list: std::unordered_map<size_t, Call>
+ CallManager()
+ ~CallManager()
+ create(): Call
+ delete(size_t idCall): bool
Use
Call
- id: size_t
- account: std:: unordered_map <size_t, CallTunnel>
+ Call()
+ ~Call()
+ addAccount(char[32] loginAccount, CallTunnel callTunnel): bool
+ deleteAccount(char[32] loginAccount): bool
Use
CallTunnel
- ip: std::string
- port: u_int32
+ CallTunnel()
+ ~CallTunnell()


Server
- _port: unsigned int
- _networkManager: std::shared_ptr<INetworkServerr>
- _handleCmd: HandleCmd
- _accountManager: AccountManager
- _callManager: CallManager
+ Server(unsigned int port)
+ ~Server()
+ run(): void
+ getLogInTerm() const: const LogInTerm &
+ getAccountManager(): AccountManager &
+ getAccountManager() const: const AccountManager &
+ getNetworkManager(): std::shared_ptr<INetworkManager>
Use
Use
Use
Use
Use
LogInTerm
_printTypeFunc: std::unordered_map<LevelLog, std::function<void(const std::string)>
+ LogInTerm();
+ ~LogInTerm();
+ print(const std::string & msg, const LevelLog level = LevelLog::DEFAULT) const: vo


HandleCmd
- cmdList: std::unordered_map<std:uint32_t, std::function>
+ HandleCmd()
+ ~HandleCmd()
+ execCmd(size_t tunnelId, std::uint32_t code, char data[B_NETWORK_BUFFER_SIZE]): bool
<<Interface>>
ICmd
+ runCmd(std::vector<char> data): bool
1 - SignUp
2 - LogIn
3 - DelContact
4 - AddContact
5 - Leave
6 - GetContact
7 - InviteCall
8 - LeaveCall
9 - JoinCall


<<Interface>>
INetworkManager
+ virtual acceptClient(): void
+ virtual write(size_t tunnelId, dataToWrite data): bool
+ virtual write(std::string login, dataToWrite data): bool
+ virtual writeForAll(dataToWrite data): bool
+ virtual getTunnelInfoByTunnelId(const size_t tunnelId): TunnelInfo
NetworkManager
- _server: Server
- _port: unsigned int
- _tcpServer: NetworkTcpServerBoost
- _tunnelInfo: std::unordered_map<size_t, TunnelInfo>
- _tunnelList: std::unordered_map<size_t, NetworkTcpServerTunnelBoost::pointer>
+ NetworkManager(babel::Server &server, unsigned int port)
+ ~NetworkManager()
+ virtualacceptClient(): void
+ virtual write(size_t tunnelId, dataToWrite data): bool
+ virtual write(std::string login, dataToWrite data): bool
+ virtual writeForAll(dataToWrite data): bool
+ virtual getTunnelInfoByTunnelId(const size_t tunnelId): TunnelInfo
+ addTunnel(NetworkTcpServerTunnelBoost::pointer tunnel): void
+ removeTunnel(NetworkTcpServerTunnelBoost::pointer tunnel): void
Use
Use
NetworkTcpServerBoost
- _ioService: boost::asio::io_service
-_server: Server &
- _networkManager: NetworkManagerBoost &
- _port: unsigned int
- _threadIoService: boost::shared_ptr<boost::thread>
- _acceptor: boost::asio::ip::tcp::acceptor
+ NetworkTcpServerBoost(Server &server, NetworkManagerBoost &networkManager, unsigned int
+ ~NetworkTcpServerBoost()
+ waitClient(): void
+ runIoServer(): void
NetworkTcpServerTunnelBoost
Header: struct
std::uint32_t: _actionCode
std::uint32_t:_dataSize
- _server: Server &
- _socket: boost::asio::ip::tcp::socket
- _headerRead: Header
- _dataRead: char [B_NETWORK_BUFFER_SIZE];
- _tunnelId: size_t
+ NetworkTcpServerTunnelBoost(Server &server, boost::asio::io_service& io_service);
+ write(dataToWrite data): void
+ close(): void
+ getTunnelId() const: const size_t &
+ create(Server &server, boost::asio::io_service& io_service): pointer
+ socket(): boost::asio::ip::tcp::socket&
- readHeader(): void
- readData(): void
- handleHeaderRead(const boost::system::error_code& error): void
- handleDataRead(const boost::system::error_code& error): void
- handleWrite(const boost::system::error_code &error): void