Algo-trading market client HLD

Terminology

(Copied from the market server HLD)

**Commodity**

A traded resource, can be held by the user or by the other actors and can be sold/bought.

**Funds**

Money being held by the user, used to purchase commodities.

**Request**

An order or a query given to the market by the user. For example: A ***sell request*** is an order given to the market telling it to sell a commodity for funds. A ***buy request*** is an order given to the market telling it to buy a commodity using funds.

Sell request

A sell request is a request that when posted to the server shows the will of a user to sell some amount of a commodity for a certain price. Posting sell requests automatically reduces the amount of available commodity the user has.

A sell request contains a commodity id, amount and ask price. If the user has enough resources to make the sell, the market replies with a sell ID.

Buy request

A buy request is a request that when posted to the server shows the will of a user to buy some amount of a commodity for a certain price. Posting buy requests automatically reduces the amount of available funds the user has.

A buy request contains a commodity id, amount and bid price. If the user has enough resources to make the sell, the market replies with a buy ID.

Cancel request

A cancel request contains an ID of the sell/buy request to cancel and refunds any commodities or funds invested in that request.

Query sell/buy request

This request returns information about the buy/sell request with the given id.

The query request contains an ID of the sell/buy and returns the current information about the sell/buy

Query user request

The query request is empty (besides authentication) and returns sell/buy IDs, and for each commodity, how many the user owns (zero if the user doesn’t own it)

Query market request

This request returns information about the market state of a certain commodity.

The query contains a commodity ID and returns the best ask price and best bid price for that commodity in the market.

**Ask**

The price on a sell request existing in the market.

**Ask price**

The lowest ask for a certain commodity out of all of the sell requests.

**Bid**

The price on a buy request existing in the market.

**Bid price**

The highest bid for a certain commodity out of all of the buy requests.

3 Tier Design

The application will follow the 3-tier design structure:

* Presentation Layer
* Business Layer
* Data Layer

Each layer will have its own namespace or project.

Presentation Layer

There are two options for UI:  
**# GUI  
# CLI**

**GUI**

The GUI allows the user to have an easy and intuitive interface.   
The design focuses on simplicity and "**3 clicks principle**" (the user wouldn't need to click on more than 3 button in order to perform an action).  
  
The main window contains buttons other forms or display a message to the user, for each action.   
Buttons in the main window are divided into 4 categories for quick distinction and organization:  
Market actions

Here the user will be able to send buy/sell requests and cancel existing requests.

Information

Here the user will be able to send queries to the server about the user’s own assets, ask and bid prices of commodities and status of existing requests.

Default AMA configuration

Here the user will be able to activate and deactivate the Autonomous Market Agent.

Activation will disable the rest of the GUI to avoid synchronization issues. Deactivation will re-enable the GUI.

User AMA configuration

Here the user will be able to configure their own Autonomous Market Agent based on new rules and conditions.

Activation will disable the rest of the GUI to avoid synchronization issues. Deactivation will re-enable the GUI.

**CLI (Legacy interface)**

The CLI is the command line interface. This is a legacy build from previous milestones. It does not contain the latest features and serves as an alternative in case of issues with the GUI during development.

Business Layer

**Server Communications**

All the data is stored and processed in the Market Server. Control over the user's assets is done by sending requests for market actions to the Market Server. The business layers receives requests from the user via the GUI and returns the responses received by the internal process of the server.

Any requests sent and responses received will be logged in a history file. The **history** file will not make a distinction between **user** and **AMA** and is included in the **Data Layer**.

**Autonomous Market Agent**

The market client will hold an AMAthat is designed to intelligently operate without user input. The agent queries data on the market, evaluates it and acts accordingly with the goal of making profit.

The user is able to configure a separate AMA

The AMA will not operate parallel to the user. Activation of the AMA will block the user from sending requests.

### **Request** **Frequency**

Due to server restrictions - the AMA will send at most 20 requests every 10 seconds.

Data Layer

**History**

All requests sent and their responses are recorded in a history file.

The history is viewable inside the GUI.

**Logs**

The program has a log that documents all actions performed.