Tibra Capital - Simple Stock Exchange

NITISH KANABAR
nkanabar@gmail.com

Design

The StockExchange class implements the IStockExchange interface. The callbacks are implemented using `boost::function`.

The list of supported stockCodes is stored in a static data-member that is initialized when the stock-exchange object is created. This data-member should support very fast lookup and I have used the `STL::set` for this lookup. Another appropriate data-structure would have been the `STL::hash_set`.

The Order class implements the interface for order objects. When the exchange receives an order, it first validates the attributes of the order and then creates an order object. The order objects are maintained in a lookup keyed with the exchangeOrderId. This lookup is implemented using `STL::map`. Removing an order from the exchange involves removing the appropriate order object from this lookup.

Feeds
The BestPriceFeed class implements the functionality for maintaining the best-price feed. Detailed notes regarding implementation decisions are in the header `BestPriceFeed.h`.

Tests
Tests are implemented using `Boost.Test` in `Test.cpp`. These tests cover the functionality of the exchange. Extensive testing is done to ensure the correctness of the BestPriceFeeds.

Improvements

- Call-backs for Error conditions.

  OnOrderAdded and OnOrderRemoved are triggered once for each call to AddOrder and RemoveOrder respectively. The call-backs functionality can be made more efficient by limiting the call-backs to trasmit error conditions.

  AddOrder can be modified to return the exchangeOrderId. If an error occurs, AddOrder returns a negative value and the exchange calls OnOrderAddError to allow the client to handle the error condition.

  Similarly RemoveOrder triggers OnOrderRemoveError when the exchange fails to remove the specified order.

  Limiting the call-back to error conditions will result in fewer call-back calls and better order throughput.

- Order Types - the interface can be extended to support different order-types such as limit orders, market orders, etc.

- Order Matching - the interface can be extended to support order matching as well as different matching algorithms.