



Technical Specifications

Basket Execution Service

Description

The term Basket Trade is a generic term for trades executed as a group. It can be thought of as a shopping basket of securities. A good example of this would be a mutual fund. A mutual fund may have the following composition:

20% Stock A
20% Stock B
60% Stock C

If you purchased 100 shares of this mutual fund, the fund manager would have to execute a basket trade to buy 20 shares of A, 20 shares of B and 60 shares of C in order to sell you 100 shares of the mutual fund.

Baskets need not be just purchases. A basket definition may be:

Buy 200 Stock A
Buy 20 Stock B
Sell 30 Stock C
Sell 150 Stock D

Buying this basket means buying A and B, and selling C and D. Selling this basket means selling A and B, and buying C and D.

Another example of a basket is a tracking basket that acts like something else. Example: Assume that the above basket tracks the NASDAQ, but lags 15 minutes behind, i.e. if the NASDAQ goes up the basket goes up 15 minutes late. In this case, the trader can watch the NASDAQ and if it goes up, the trader can buy the basket and expect it to go up when 15 minutes has passed, then sell the basket for a profit.

Basket trades are complex, however, because the basket is composed of multiple instruments that must be bought or sold as a unit. Even if all the orders are placed in the market at the same time, there is no guarantee that the orders will be filled at the same time. Furthermore, the market can move dramatically between the time the first order is executed and the final order is executed. In the above example, if the market were going down it would be bad if you bought A and B, while the C and D sell orders sat until the market bottomed out. This would be buying high and selling low within the basket.

The Basket Execution System mitigates some of these risks by allowing the traders to balance the orders. A portion of the Buy and Sell orders are placed and executed as a group, minimizing the risk of market movement.

Overview

The basket execution server provides a way to execute baskets of size one to n . Basket orders received from various clients are evaluated, monitored and executed in a balanced manner based on user input and market data from the TIB/Wombat data feeds. Order flow is routed via the OM, and executions are entered into the TD securities database.

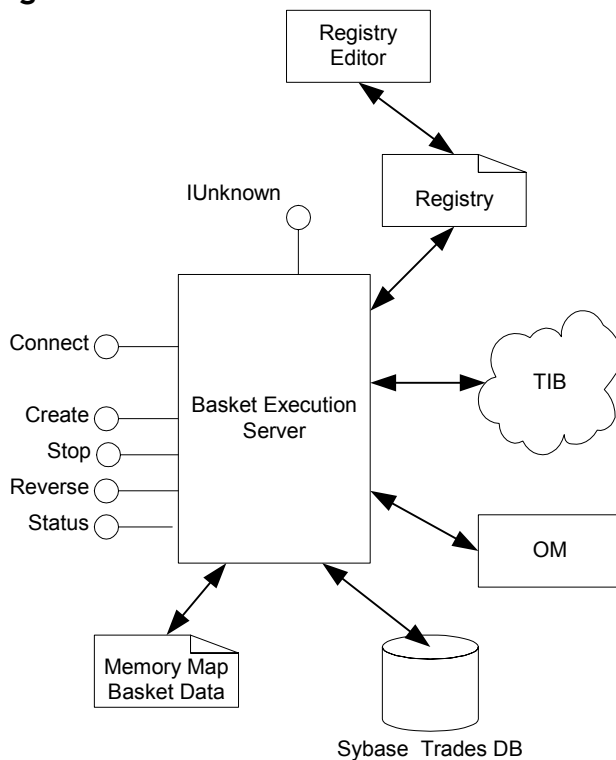
This project results in the creation of a basket execution server with a COM/DCOM interface. The interface includes methods for creating, reversing, stopping, and retrieving the status of basket orders. The server is constructed such that multiple servers can run concurrently on a single box. The server persists basket orders in a memory-mapped file for robustness. Executions are stored in the existing Sybase database, and periodic status of basket orders are published to the TIB.

Project Detail

Technology

- COM/DCOM API, targeting Windows based front-end clients.
- MSMQ to allow for multiple basket execution servers to run on a single box.
- Memory Mapped file for order persistence.
- Internal systems: OM, TIB, Sybase

Logical Architecture



DCOM API: Utilizing a DCOM API, the basket execution service can be more easily integrated into Windows based front-end applications, including Excel and VB

applications. Before the server can be accessed, the application must first execute the Connect () method to connect to a specific server as well as authenticate.

Order Persistence: For robustness, basket orders must be persisted in case of a service crash or other problem. The data could be persisted in a database, if one is available. This document assumes that one is not available. The target platform for this service is a Windows 2000 server. Memory mapped files provide an efficient means for persisting data. The service views the file as if it were memory, and the Windows operating system handles the writing of data to the memory mapped file.

External Services: This service interacts with the order routing system, Customer's Sybase database for trades, and the TIB infrastructure. The market data required by the Basket Execution Service comes from the TIB/Wombat feeds. The TIB is also used as a basket status reporting mechanism, as the basket execution service publishes periodic status updates to the TIB. The market for all open orders is monitored to determine which ECN or exchange an order should be routed to. Orders may be cancelled and reopened if the best market moves from one ECN or exchange to another. Order flow is handled via the OM. Order executions are inserted into the existing trades database. As with previous projects, AURA Consulting, Inc. accesses the trade database via stored procedures built by Customer.

Platform: The target platform is a Windows 2000 box. The server is written such that multiple servers can run on the same box simultaneously. The API must, however, have the ability to connect to one of several basket servers that may be running simultaneously on different boxes.

Business Logic

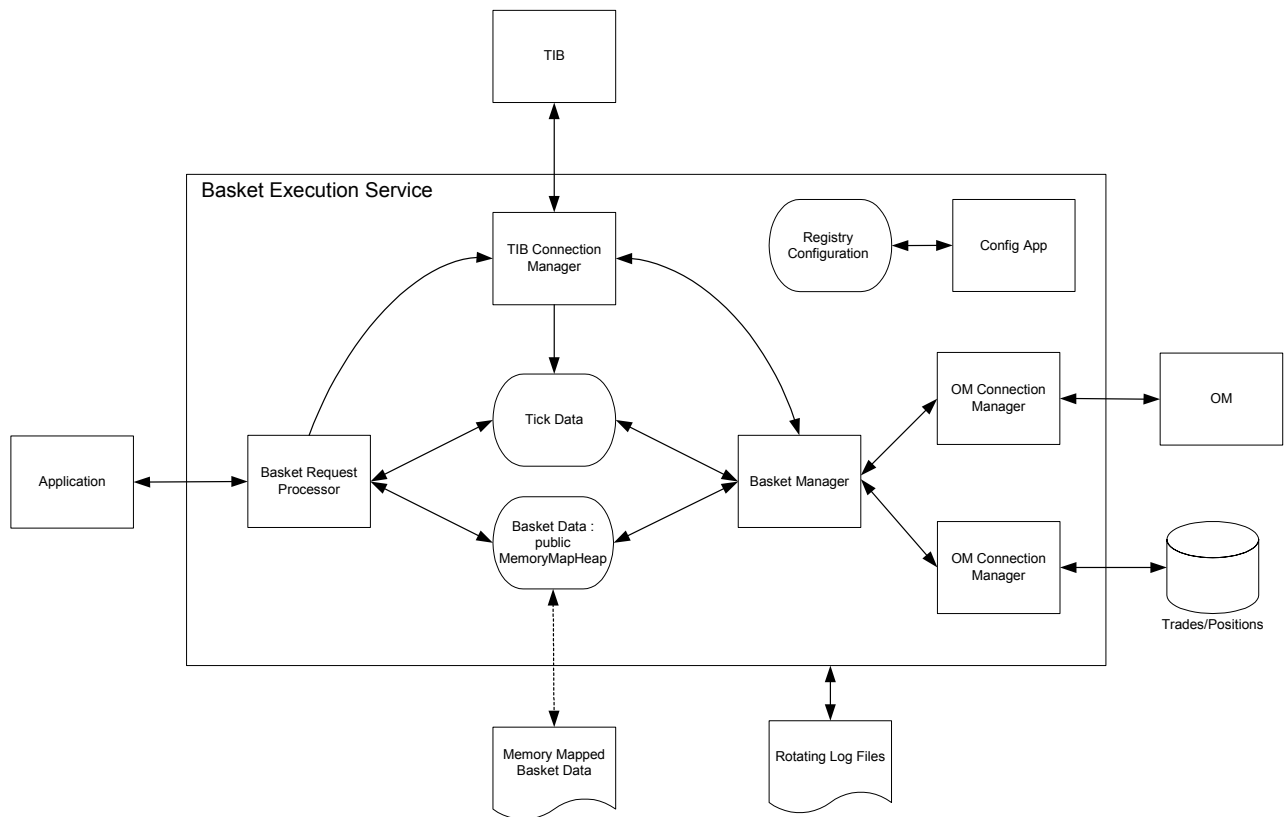
- **Basket Orders:** Inputs required to create a basket order are:
 - i. The Basket execution server interface accepts an array of symbols or symbol pointers to specific ECN's to define the instruments to be traded.
 - ii. Quantity per stock
 - iii. Min/Max basket price, or Min/Max long value to short value ratio
 - iv. Execution timeout
 - v. Failure strategy (stop current execution, reverse executions)
 - vi. Aggressiveness level (To define the selection of Market, or Bid/Mid/Ask Limit orders for the components of the basket)
 - vii. Basket balancing level
- **Balanced Executions:** Basket orders are processed in a balanced way (between long and short orders) to reduce market exposure. Short sales are opened first, and longs are opened as the shorts are filled. The basket balancing level indicates the desired short/long percentage. 0 indicates no balancing, 100 strict/complete balancing. The exact balancing algorithm was defined at the beginning of the project.

- Order Status: Two mechanisms are provided for basket status. A new message is created for publishing basket status to the TIB infrastructure. Also, the service API provides a mechanism to retrieve the status. TIB status updates include the following message types:
 - i. Order Acknowledgement: order received
 - ii. Status Update with the state of all orders in the basket, including basket ID, stock, shares filled, shares remaining, shares total, average price.
 - iii. Completed: basket order is in a completed state. No more action will be taken
 - iv. Stop: open orders canceled, no more action will be taken
 - v. Reverse: order reversal commencing.
 - vi. Failure: failure strategy commencing.
- Access Restriction: The exact mechanism to restrict access to the service was determined. Authentication is performed in the Connection () method.

Risk Factor

- Security: The security mechanism was defined. The initial project proposal did not allocate time for a complex security mechanism. However, a simple session logon was possible given database resources, or an existing account database.
- Load: Monitoring live ticks is an issue if the volume is great. The number of open orders is limited to prevent information overload.
- Multiple Services On One Host: With a DCOM implementation, MSMQ is needed to route messages to the different instances running on a given host. This adds some complexity and time to the project.

Architecture



Acceptance Criteria:

- BEP handles baskets of size 1 to 2000, with an average basket size of 10
- BEP capacity is up to 10,000 concurrent baskets for any one instance, consisting of at most 5,000 different stocks, listening to an average of 3 ECN's.
- BEP handles one-sided baskets (i.e. all buys or all sells).
- Status messages are published upon each and every leg of basket being filled.
- Any basket that is completed or not reversed completely is persisted such that a reversal message accompanied by a basket ID triggers a reversal of only those stocks that have fills/partial fills.
- A mechanism to output the state of all baskets at any time via a specific tib message was provided.
- A configurable time limit (in terms of days) for persistent basket data was included.
- A log file is generated with a date stamp in its name that logs the following:
 - Orders Received
 - Trades Placed
 - Fills/partial fills
 - Cancels/Replaces placed
 - Cancels/Replaces completed

