# PORTFOLIO EXECUTION AND REPORTING

## CROSS-REFERENCE TO RELATED APPLICATIONS

**[0001]** This application claims priority from and claims the benefit of U.S. Provisional Application No. 60/677,963, filed May 5, 2005, entitled "Portfolio Execution and Reporting Facility", which is hereby incorporated by reference.

## BACKGROUND

**[0002]** Portfolio traders manage portfolios of stocks and/or other financial instruments and, when they trade, they often times make large dollar value trades in a number of different stocks or other instruments. Portfolio traders are typically fund managers or traders and brokers acting on behalf of a large institution, such as a mutual fund or a money-management firm. Portfolio traders employ a variety of different trading strategies to manage their portfolios, and they use program trading to implement these strategies.

**[0003]** A program trade is a series of related purchases or sales of a group of securities where the related purchases and sales satisfy parameters for a minimum number of securities and a minimum market value. The specific parameters of a program trade are defined by market regulations, which currently define a program trade as the purchase or sale of a basket of at least fifteen securities with a total market value of at least one million dollars.

**[0004]** Several limitations exist in prior systems in which program trade orders are executed. For example, in some prior systems, where the portfolio trader executes his program trade orders on an exchange in the market place, the portfolio trader is required to execute the orders during the trading day. If the size of the trade print is large enough, executing this order during the trading day will likely have the negative effect of moving the market away from the position of the portfolio trader that

executed the order. This is due to the fact that trades executed within the marketplace are reported immediately. Furthermore, program trade orders executed on an exchange during the trading day run the risk of being broken up due to interaction with the marketplace. Such market interaction poses a serious issue for a trader because a trader wants the program trade orders that make up the program trade to each cross cleanly at a single price. With market interaction, an individual program trade order that is part of the program trade gets separated into a matched portion that crosses and an unmatched portion which must be executed manually. The result is that the individual program trade order ends up getting traded, potentially, at multiple prices.

**[0005]** To remedy this problem, portfolio traders often execute such program trade orders off the exchange (*e.g.*, at an institutional trading desk or within an alternative trading system such as a crossing network). While this method resolves the trader's problem by keeping large orders from interacting with the marketplace and from having execution of the order reported out immediately and potentially negatively moving the market, such order executions must still be reported. Marketplace rules require that executed trades be reported to an exchange or comparable public market center. Program trade orders that are executed off an exchange typically are reported either at the end of the day as aggregated program trade totals, with the details of the execution being reported over the next several days, or reported the next morning as individual trades.

**[0006]** Crossing systems have been developed where program trade orders may be executed on a public exchange, and execution of such orders is captured for market reporting. For example, the New York Stock Exchange ("NYSE") offers a special

trading session (*i.e.* Crossing Session II) where program trade orders may be received and executed on the NYSE at the end of the day, after the market has closed.

**[0007]** Accordingly, there is a need for a crossing system that allows traders to submit their program trade orders throughout the trading day at the trader's convenience for execution at a pre-specified time so that these orders do not interact with other orders on the exchange.

## SUMMARY

**[0008]** According to an aspect of the present invention, a method for processing a program trade, includes providing a crossing market center and designating a portfolio crossing session start time outside of normal market trading hours and throughout the trading day until the designated portfolio crossing session start time, receiving and storing a plurality of program trade orders at the crossing market center, wherein each program trade order contains a basket identification number identifying the program trade that the program trade order is part of. It further includes, at the designated start time, initiating a portfolio crossing session and executing the received program trade orders, resulting in portfolio crosses. It further includes linking together the completed portfolio crosses having the same basket identification number.

## DESCRIPTION OF THE DRAWINGS

**[0009]** These and other features, aspects and advantages of the present invention will become better understood with regard to the following description, appended claims and accompanying drawings where:

**[0010]** FIG. 1 is a block diagram illustrating the trading environment in which an embodiment of the present invention operates;

**[0011]** Fig. 2 illustrates the process implemented by a portfolio cross order implementation program when the program receives a portfolio cross order; and

[0012] FIGS. 3A-3B illustrates a process implemented by an embodiment of the present invention in performing surveillance on the trades executed through the system.

## DETAILED DESCRIPTION

[0013] Referring to Fig. 1, a trading environment in which an embodiment of the system and method of the present invention operates is depicted. The examples discussed herein describe the use and application of the present invention in an equity security market center environment, but it should be understood that the present invention could be used in any type of financial instrument market center environment (*e.g.*, securities, futures contracts, options, bonds, etc.). The trading environment of this embodiment includes a crossing market center 20 which interacts with a number of other market centers 24 (*i.e.* away market centers) and traders 26. It should be understood that the crossing market center 20 referred to herein refers to a computing system having sufficient processing and memory capabilities and does not refer to a specific physical location. In fact, in certain embodiments, the computing system may be distributed over several physical locations. It should also be understood that any number of traders 26 or away market centers 24 can interact with the crossing market center 20. The crossing market center 20 is the market center to which a portfolio trader 26 sends a specific program trade order. The crossing market center 20 includes an order matching engine 21, which validates, matches and processes all orders on the market center 20. In this embodiment, the order matching engine 21 includes a program trade order implementation program 22, which functions to facilitate the execution of program trade orders sent to the crossing market center 20. The program trade order implementation program 22 may also be utilized as stand alone code separate and apart from the order matching engine 21. In this

embodiment, the code for the order matching engine 21 and for the program trade order implementation program 22 are stored in the crossing market center's memory.

**[0014]** The crossing market center 20 may also include a program trade order parameters data structure 27. The program trade order parameters data structure 27 stores pre-defined parameters and rules that are used by the order matching engine 21 in executing program trade orders (*e.g.*, portfolio crossing session start time, program trade order definition parameters such as minimum number of symbols in a portfolio and minimum portfolio value parameters, etc.). The crossing market center 20 may also include an order and execution interface 28 which interacts with the traders 26, the away market centers 24 and the order matching engine 21 in crossing a program trade order and, in this embodiment, the regular order execution process. The crossing market center 20 may also include a program trade order data structure 29 where program trade order information is stored, an order execution data structure 30 where executed order information is stored and a trade reporting data structure 31 where trade reporting information is stored (*e.g.*, trade reporting rules and the tape data for trade reports).

**[0015]** Throughout the discussion herein, it should be understood that the details regarding the operating environment, data structures, and other technological elements surrounding the crossing market center 20 are by way of example and that the present invention may be implemented in various differing forms. For example, the data structures referred to herein may be implemented using any appropriate structure, data storage, or retrieval methodology (*e.g.*, local or remote data storage in data bases, tables, internal arrays, etc.). Furthermore, a market center of the type described herein may support any type of suitable interface on any suitable computer system.

Incoming Program Trade Order

**[0016]** Referring to Fig. 2, at step 100, a new program trade order is received by the order matching engine 21. The order matching engine 21, recognizing the program trade order designation, stores the program trade order to the program trade order data structure 29 and queues this order for later execution, as indicated at step 102. The program trade order has an order identification number and a basket identification number that are assigned by the trader 26 that sent the order. The order identification number acts as a reference and tracking identifier for that order, and the basket identification number is a reference which links all of the orders in the portfolio basket of crosses together. Since program trade orders are created so that they do not need to interact with the general market, price/time priority rules, from an order matching standpoint, are irrelevant. As every order is guaranteed to execute at its specified order price, the process simply stores the received program trade orders for each symbol in the order in which such order was received. The process can optionally store the orders according to symbol or basket identification number or can simply store all the received orders sequentially in a single file. As each order represents a self-contained cross, the sequence in which the orders are stored is irrelevant. The stored program trade orders, in this embodiment, are not displayed on the crossing market center's order book or to the general marketplace in any manner, which keeps the trader's trading intentions from the market.

**[0017]** The portfolio crossing time parameter (*e.g.* "Portfolio Crossing Time") is a pre-set parameter which sets the time that the daily portfolio crossing session commences and resides in the program trade order parameters data structure 27. As indicated at step 110, at the point in the day when the portfolio crossing time parameter is reached, the process initiates a portfolio crossing session, as indicated at

step 112. Typically, the portfolio crossing session is set up to execute outside of normal market hours to keep the portfolio crossing orders from interacting with the market. However, it should be understood that the start time for the portfolio crossing session can be set for any time (*e.g.* set for prior to market opening rather than after the close of the market) and that the process can be set up to conduct more than one portfolio crossing session in a day if desired (*e.g.* one prior to the market opening and one after the close of the market). Also, it should be understood that, in this embodiment, submitted program trade orders stored on the program trade order data structure 29 may be modified and/or canceled prior to the commencement of the portfolio crossing session.

**[0018]** The portfolio crossing session commences by retrieving a program trade order stored in the program trade order data structure 29, as indicated at step 114. At step 116, the process retrieves the Trade Reporting rule for the symbol in the program trade order record. At step 122, the process executes the cross defined by the program trade order (*i.e.* price and size).

Reporting the Executed Program Trade Crosses

**[0019]** In this embodiment, as each program trade order is executed, each resulting cross is reported. It is contemplated that in other embodiments the crosses might not be reported. Reporting requirements for crosses are based on the symbol traded. When the process reports on an executed cross, it retrieves the reporting rules that apply for that symbol from the trade reporting data structure 31 and reports on the trade in accordance with those rules. For example, one market center or Trade Reporting authority, away market center 24a, may require crosses to be reported in aggregate and that crosses be reported immediately after the end of the portfolio crossing session; while another market center or Trade Reporting authority, away

market center 24b, may require the crosses to be reported individually for each symbol and that they be reported the next morning. If the crosses are to be reported in the aggregate, the process computes the sum of all the shares and the total value of all the shares that were crossed during the portfolio crossing session, combining together all crosses for all traders for the market center symbols being aggregated.

**[0020]** Referring again to Fig. 2, at step 126, the process determines how the symbol executed in the cross is reported, specifically it determines whether the symbol crossed is reported to the tape in aggregate. If the symbol is not to be reported in the aggregate, the cross is reported individually to the tape, as indicated at step 128, and the process proceeds to determine if there are any additional stored program trade orders that need to be executed, as indicated at step 136. If the process determines that the symbol crossed needs to be reported in the aggregate, then, as indicated at step 130, the process adds the volume of shares of the symbol crossed to the running total of aggregate shares that have been crossed in the trading session to that point for all symbols for a specific market center *(i.e.* parameter AggregateSharesCrossed stored on the trade reporting data structure 31). At step 132, the process determines the total value of the aggregated shares by multiplying the volume of the cross by the price per share of the cross. Then, at step 134, the process adds the computed trade value to the running total of aggregate value of the shares that have been crossed in the trading session to that point for all symbols for a specific market center *(i.e.* parameter AggregateValueCrossed stored on the trade reporting data structure 31). At step 136, the process then determines if there are any other stored program trade orders that need to be executed. If yes, then the process returns to step 114 and retrieves the next program trade order and processes the order in the manner described above. If, at step 136, it is determined that there are no further program trade orders

that need to be executed, then the process reports the value of the aggregate shares crossed parameter and the aggregate volume crossed parameters to the tape, as indicated at step 138. The portfolio crossing session is then complete, as indicated at step 140.

Surveillance of Program Trade Crosses

**[0021]** Only orders that meet the definition of a program trade order are eligible to be treated and executed as a program trade order. In this regard, the process of the present invention checks the program trade orders executed to verify that the basket of crossed orders did in fact meet the definition of a program trade, which, in this embodiment, is a basket of crossed orders having a value of at least one million dollars and including at least fifteen securities' symbols. In this embodiment, the process validates a basket of crossed orders after execution. However, it is contemplated that in other embodiments a basket awaiting execution could be validated prior to execution.

**[0022]** Referring to Figs. 3A-3B, after completion of the portfolio crossing session, order surveillance takes place. At step 142, the process initiates the portfolio crossing surveillance reporting process. At step 144, the process retrieves the parameters for a valid program trade order (*i.e.* the parameter which defines the minimum number of symbols that need to be included in the basket ("MinPortfolioSymbols") and the parameter which defines the minimum dollar value for the basket ("MinPortfolioValue"). In this embodiment, these parameters are retrieved from the program trade order parameters data structure 27.

**[0023]** At step 146, the process retrieves all the program trade crosses executed during the portfolio crossing session and sorts these crosses by basket identification number. At step 148, the process retrieves a program trade cross record. If the record

retrieved is the first record, the basket-in-process parameter is set to the basket identification number of the retrieved record, as indicated at step 150. At step 152, the process determines whether the basket identification number for the record retrieved is the same as the basket-in-process parameter. If the two values are the same, then the process proceeds to step 154, where the process adds to the parameter that counts the number of symbols in the portfolio basket ("PortfolioSymbolCount"). Then at step 156, the process computes the value of the cross for the record being analyzed which is equal to the cross volume multiplied by the price per share. At step 158, the process keeps a running total of the entire value of the portfolio crossed by adding the value computed for the individual cross at step 156 to the total value at that point. Referring back to step 152, if the basket identification number being processed is not the same as the basket-in-process parameter, the process proceeds to step 162 where, in this embodiment, it prints out the basket identification number for the basket that was just analyzed. It also prints out the total number of symbols that were in the portfolio basket, and it prints out the total value of the crosses that were executed in the basket. At step 164, the process checks to make sure the portfolio basket included the proper number of symbols for a program trade. The process does that by determining whether the number of symbols that crossed within the basket ("PortfolioSymbolCount") is greater than or equal to the parameter that defines the minimum number of symbols that must be present in a portfolio basket ("MinPortfolioSymbols"). If the number of symbols crossed in the portfolio basket is less than the required minimum, the process, as indicated at step 166, prints an error message which indicates that the basket did not contain the requisite number of symbols. If the basket did have at least the minimum number of symbols, the process continues to step 168 where it determines whether the basket of executed crosses

satisfied the requisite dollar value as defined by the minimum basket value parameter. If the value of the basket of trades is greater than or equal to the minimum basket value parameter, then the criteria for the minimum basket value has been satisfied,, and the process proceeds to step 172. However, if the value of the basket of crosses executed is less than the minimum basket value parameter, the process, as indicated at step 170, prints an error message which indicates that the basket did not meet the minimum value required for a program trade. At steps 172 and 174, the process, re-sets the field for the total number of symbols in the basket to zero and re-sets the field for the total value of the basket of crosses to zero, respectively. At step 176, the process then sets the basket-in-process parameter equal to the basket identification number for the basket now being analyzed and proceeds to step 154.

**[0024]** At step 160, the process determines if there are remaining program trade crosses to analyze. If yes, the process returns to step 148 and repeats the process described above for the next program trade cross record. If there are no further program trade crosses to analyze, then the portfolio crossing surveillance process is complete, as indicated at step 178.

**[0025]** Examples of program trade orders received by a crossing market center 20 are provided below. It should be understood that the order prices discussed in the examples below are by way of example only to illustrate how the process of an embodiment of the invention handles program trade orders of the present invention. For ease of illustration in showing how different symbols have different Trade Reporting requirements, the orders in the examples below have been aggregated according to symbol. As previously described, however, the orders may be stored in any sequence required.

A. Example Incoming Program Trade Orders

**[0026]** As steps 100 and 102 indicate, program trade orders sent to the crossing market center 20 throughout the day are captured and stored as a record in the program trade order data structure 29.

Example A1: Program Trade Order

**[0027]** In this example, the crossing market center 20 receives the following program trade order from trader 26a:

→ Order 17: Cross 10,000 Symb01 @ 20.04, Portfolio Cross, BasketID = A127

**[0028]** The record for this order stored in the program trade order data structure 29, in this example, has the following format:

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb01 | Trader26a | 08:05:13 | 17 | A127 | 10,000 | 20.04 |

**[0029]** Trader 26a has assigned order identification number 17 and basket identification number A127 to this order. The basket order identification number links this order to all of the other orders with a basket order identification number of A127. All of the orders linked together in "basket" A127 constitute the portfolio being traded.

Example A2: Program Trade Order for the Same Symbol, But Different Trader

**[0030]** In this second example, the crossing market center 20 receives the following program trade order from a different trader 26b:

→ Order 202: Cross 9,200 Symb01 @ 20.05, Portfolio Cross, BasketID = B1743A

The queue of records for Symb01 now appear as follows:

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb01 | Trader26a | 08:05:13 | 17 | A127 | 10,000 | 20.04 |

| | | | | | | |
|---|---|---|---|---|---|---|
| Symb01 | Trader26b | 08:40:35 | 202 | B1743A | 9,200 | 20.05 |

Example A3: Program Trade Order for the Same Symbol, But Different Basket

**[0031]** Traders can submit multiple program trade orders for the same symbol if the symbol is a constituent of more than one basket. In this third example, the crossing market center 20 receives the following program trade order from the first trader 26a for symbol Symb01, but for a different basket:

→ Order 103: Cross 16,000 Symb01 @ 20.10, Portfolio Cross, BasketID = F234

**[0032]** This program trade order is inserted into the queue for symbol Symb01. None of these orders for Symb01 will ever interact with each other. By definition, program trade orders do not interact with any other orders.

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb01 | Trader26a | 08:05:13 | 17 | A127 | 10,000 | 20.04 |
| Symb01 | Trader26b | 08:40:35 | 202 | B1743A | 9,200 | 20.05 |
| Symb01 | Trader26a | 09:33:24 | 103 | F234 | 16,000 | 20.10 |

Example A4: Program Trade Order for Different Symbol for an Open Basket

**[0033]** In this example, traders must submit cross orders for at least 15 different symbols for the same basket. In this fourth example, the crossing market center 20 receives the following program trade order from the first trader 26a for symbol Symb02 for an open basket, basket A127:

→ Order 268: Cross 20,000 Symb02 @ 17.57, Portfolio Cross, BasketID = A127

The process inserts the order in the queue for symbol 'Symb02':

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb02 | Trader26a | 10:17:40 | 268 | A127 | 20,000 | 17.57 |

B. Example Portfolio Crossing Session

**[0034]** In this example, the late trading session ends on the crossing market center 20, and the market closes. In this example, away market center 24a and away market center 24b have the following tape reporting requirements for program trade orders executed on the crossing market center:

- Crosses in the symbols 'Symb01' and 'Symb03' must be reported to away market center 24a. Away market center 24a requires program trade crosses to be reported in aggregate form at the end of the portfolio crossing session; and
- Crosses in the symbol 'Symb02' must be reported to away market center 24b. Away market center 24b requires program trade crosses to be reported individually at the beginning of the next trading day following the close of the portfolio crossing session.

**[0035]** Referring to step110, in this example, shortly after market close, the time equals the time designated by the portfolio crossing time parameter. As indicated at step 112, the portfolio crossing session is initiated.

Example B1: First Portfolio Cross in Symb01 Executes

**[0036]** The queue for symbol 'Symb01' looks like this when the portfolio crossing session commences:

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb01 | Trader26a | 08:05:13 | 17 | A127 | 10,000 | 20.04 |
| Symb01 | Trader26b | 08:40:35 | 202 | B1743A | 9,200 | 20.05 |
| Symb01 | Trader26a | 09:33:24 | 103 | F234 | 16,000 | 20.10 |

**[0037]** As indicated at steps 114 and 116, the process retrieves the first program trade order and the Trade Reporting rules for the symbol designated therein (*i.e.* Order

17 and Symb01). At Step 122, the process executes the order, crossing 10,000 shares of Symb01 at $20.04:

→ Trader26a Crossed 10,000 Symb01 at $20.04

**[0038]** Then, at step 126, the process determines whether crosses for Symb01 are reported in the aggregate or not. In this example, since Symb01 is reported to away market center 26a, the cross needs to be reported in the aggregate. At step 130, the process adds the cross volume (*i.e.* 10,000 shares) to the computed variable AggregateSharesCrossed. In step 132, the process computes the value of the cross (*i.e.* 10,000 shares x $20.04) as $200,400. At step 134, the process then adds the value of the cross ($200,400) to the running total in variable AggregateValueCrossed. In this example, the current values of the aggregated statistics are the following:

→ AggregateSharesCrossed = 10,000

→ AggregateValueCrossed = $200,400

At step 136, the process determines that there are more program trade orders to execute and returns to step 114 to get the next order.

Example B2: Second Portfolio Cross in Symb01 Executes

**[0039]** At step 114, the process retrieves the second portfolio cross order (*i.e.* Order 202) and retrieves the Trade Reporting rules for the symbol, which, in this example, happens to be the same as the previously processed order (*i.e.* Symb01). At step 122, the process executes Order 202 crossing 9,200 shares at $20.05:

→ Trader26b Crossed 9,200 Symb01 at $20.05

**[0040]** As with the first program trade order above, at step 126, the process determines that crosses for Symb01 are reported in the aggregate. Therefore, at step 130, the process adds the cross volume (*i.e.* 9,200 shares) to the computed variable AggregateSharesCrossed. At step 132, it computes the value of the cross as

$184,460. As above, the value of the cross is added to the running total for the total value of shares crossed in this symbol. The updated aggregate values for the number of shares crossed and their value for Symb01 at this point are as follows:

→ AggregateSharesCrossed = 19,200 (10,000 + 9,200)

→ AggregateValueCrossed = $384,860 ($200,400 + $184,460)

**[0041]** Once again, the process at step 136 determines that there are more program trade orders to execute and returns to step 114 to retrieve the next order.

Example B3: Third Portfolio Cross in Symb01 Executes

**[0042]** At step 114, the process retrieves the third program trade order (*i.e.* Order 103). At step 116, the process, in this example, retrieves the Trade Report rules for Symb01 again. Then at Step 122, the process crosses 16,000 shares at $20.10:

→ Trader26a Crossed 16,000 Symb01 at $20.10

**[0043]** The process again determines that crosses for Symb01 are reported in the aggregate so it adds the cross volume (*i.e.* 16,000 shares) to the running total for the total amount of shares crossed in the specified symbol. Then, at step 132, as before, it computes the value of the cross as $321,600. In step 134, it adds the computed value of the cross to the aggregated running total of the value of crossed shares in the specified symbol. The updated values of the aggregated statistics are the following:

→ AggregateSharesCrossed = 35,200 (10,000 + 9,200 + 16,000)

→ AggregateValueCrossed = $706,460 ($200,400 + $184,460 + $321,600)

**[0044]** Since there are further program trade orders to execute, the process returns to step 114 to retrieve the next order.

Example B4: First Portfolio Cross in Symb02 Executes

**[0045]** The queue for symbol 'Symb02' looks like this at this point:

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| Symb02 | Trader26a | 10:17:40 | 268 | A127 | 20,000 | 17.57 |

**[0046]** At steps 114 and 116, the process retrieves the next stored program trade order shown above and the Trade Reporting rules for the symbol designated therein (*i.e.* Order 268 and Symb02). The process then proceeds to step 122 where it executes the order and crosses 20,000 shares at $17.57:

→ Trader26a Crossed 20,000 Symb02 @ 17.57

**[0047]** Then at step 126, as with Symb01, the process determines whether trades for Symb02 are reported in the aggregate or not. In this case, it determines they are not reported in the aggregate (Market Center 24b's rule). The process, therefore, proceeds to step 128 where it reports the cross individually to the Tape and no further processing is required. All the individual crosses that must be reported to Market Center 24b are sent to Market Center 24b the next morning per its requirements.

**[0048]** At step 136, the process determines if there are additional program trade orders that require processing. In this example, there are. The process therefore returns to step 114 and retrieves the next stored order.

Example B5: First Portfolio Cross in Symb03 Executes

**[0049]** The queue for symbol 'Symb03' looks like this at this point:

| Symbol | FirmID | Time received | OrderID | BasketID | Quantity | Price |
|---|---|---|---|---|---|---|
| Symb03 | Trader26a | 10:17:53 | 269 | A127 | 9,000 | 40.40 |

**[0050]** At steps 114 and 116, the process retrieves the next stored program trade order shown above and the Trade Reporting rules for the symbol designated therein (*i.e.* Order 269 and Symb03). The process then proceeds to step 122 where it executes the order and crosses 9,000 shares at $40.40:

→ Trader26a Crossed 9,000 Symb03 @ 40.40

**[0051]** Then, proceeding to step 126, the process determines that crosses in Symb03 need to be reported in the aggregate (Market Center 24a's rule). As such, during steps 130-134, the process continues to increment the AggregateSharesCrossed and AggregateValueCrossed parameters that were started with Symb01, even though this trade is for a different symbol. In this example, the variables AggregateSharesCrossed and AggregateValueCrossed are aggregated values for all crosses in symbols for which Market Center 24a sets the reporting requirements.

**[0052]** Therefore, at step 130, the process adds the cross volume (9,000 shares) to the computed variable AggregateSharesCrossed, and at step 132, it computes the TradeValue by multiplying the cross volume (9,000 shares) by the cross price ($40.40) to derive the TradeValue of $363,600. At step 134, it adds the TradeValue to the computed variable AggregateValueCrossed.

**[0053]** These are the updated values of the aggregate statistics in this example:

→AggregateSharesCrossed = 44,200 (10,000 + 9,200 + 16,000 + 9,000)

→ AggregateValueCrossed = $1,070,060 ($200,400 + $184,460 + $321,600 + $363,600)

**[0054]** The process continues processing stored program trade orders in this manner until all the stored program trade orders are processed. Then, at that point, as indicated at steps 136-140, the process reports the aggregated values to Tape and the portfolio crossing session is terminated. In this example, at the beginning of the next trading day, the crossing market center 20 reports the individual trades to Market Center 24b per its requirements.

C. Example of Surveillance of Program Trade Orders

**[0055]** Referring to Figs. 3A-3B, after the end of the portfolio crossing session, the process implements the portfolio crossing surveillance reporting routine, as indicated at step 142. As described earlier, in this embodiment of the invention, the

process does not perform surveillance prior to trade execution. However, it should be understood that in other embodiments, the process may be configured to check if a portfolio of cross orders satisfies the criteria for a program trade prior to the execution of the basket.

**[0056]** At step 144, the process retrieves the parameter that defines the minimum number of symbols that are required in a program trade ("MinPortfolioSymbols") from the program trade order parameters data structure 27. The MinPortfolioSymbols parameter, in accordance with market regulations, is set to 15 symbols in this example. At step 144, the process also retrieves the parameter that defines the minimum value that is required for a program trade ("MinPortfolioValue") from the program trade order parameters data structure 27. The MinPortfolioValue parameter, in accordance with market regulations, is set to $1 million in this example. At step 146, the process retrieves all crosses that were executed and sorts the crosses by basket identification number ("BasketID"). In this example, the first BasketID is BasketID=A127. These are the fifteen crosses that were executed for BasketID A127 during the portfolio crossing session:

| Time received | OrderID | BasketID | Symbol | Quantity | Price |
|---|---|---|---|---|---|
| 08:05:13 | 17 | A127 | Symb01 | 10,000 | 20.04 |
| 10:17:40 | 268 | A127 | Symb02 | 20,000 | 17.57 |
| 10:17:53 | 269 | A127 | Symb03 | 9,000 | 40.40 |
| 11:35:03 | 310 | A127 | Symb04 | 3,000 | 63.95 |
| 13:24:34 | 405 | A127 | Symb05 | 15,000 | 9.43 |
| 14:04:57 | 497 | A127 | Symb06 | 6,000 | 18.25 |
| 15:40:43 | 603 | A127 | Symb07 | 4,000 | 37.37 |
| 15:46:15 | 604 | A127 | Symb08 | 5,500 | 43.16 |
| 16:10:18 | 627 | A127 | Symb09 | 12,000 | 13.49 |
| 16:25:20 | 646 | A127 | Symb10 | 6,000 | 95.73 |
| 17:53:07 | 838 | A127 | Symb11 | 1,000 | 112.24 |
| 18:09:45 | 905 | A127 | Symb12 | 7,500 | 25.74 |
| 18:37:24 | 934 | A127 | Symb13 | 8,000 | 56.67 |
| 19:40:10 | 953 | A127 | Symb14 | 4,000 | 73.37 |
| 19:57:09 | 954 | A127 | Symb15 | 7,000 | 47.26 |

**[0057]** At step 148, the process retrieves the first cross, OrderID = 17 in this example. At step 150, since this is the first record, the process sets the parameter for the basket-in-process to A127 (*i.e.* BasketInProcess = A127). Because of step 150, the current BasketID is equal to the basket-in-process parameter. Therefore, the process moves on to step 154 where it adds one to the parameter that aggregates the number of symbols in the basket (*i.e.* PortfolioSymbolCount). At step 156, the process computes the value of the order presently being analyzed by multiplying the volume of the cross (10,000 shares) by the cross price ($20.04) to derive a cross value of $200,400. At step 158, the calculated cross value is added to the total value for the basket (*i.e.* TotalPortfolioValue).

**[0058]** After analyzing the first trade in BasketID A127, the current portfolio values are:

→ PortfolioSymbolCount = 1

→ TotalPortfolioValue = $200,400

**[0059]** At step 160, the process determines that there are additional crosses to be processed in this basket and returns to step 148 to retrieve the next cross (*i.e.* OrderID = 268). Since the BasketID for this order is the same as the basket-in-process parameter, at step 154, the process adds one to the PortfolioSymbolCount parameters. At step 156, it computes the TradeValue parameter for this cross by multiplying the volume (20,000 shares) by the cross price ($17.57) to derive a value of $351,400. At step 158, it adds the computed cross value for OrderID 268 to the aggregated TotalPortfolioValue parameter.

**[0060]** After analyzing the second trade in BasketID A127, the updated portfolio values are as follows:

→ PortfolioSymbolCount = 2

→ TotalPortfolioValue = $551,800 ($200,400 + $351,400)

**[0061]** The process continues in the manner described above for all of the remaining crosses in BasketID A127, the third through fifteenth crosses in this example. After processing all of Basket ID A127's crosses at Step 148, the process retrieves the next cross, which it recognizes as belonging to a new BasketID (B1743A) and compares it to the BasketInProcess parameter, which is presently set to A127. Since the BasketID values are not the same, the process prints the final PortfolioSymbolCount and the final TotalPortfolioValue for Basket A127 at step 162:

→ Portfolio A127

→ Symbol Count: 15

→ Portfolio Value: $3,864,270

**[0062]** At step 164, the process determines whether the PortfolioSymbolCount parameter for BasketID A127 (*i.e.* 15 in this example) is greater than or equal to the MinPortfolioSymbols parameter (*i.e.* 15 in this example). As the parameters equal each other, the process proceeds to step 168, where the process determines whether the TotalPortfolioValue for BasketID A127 (*i.e.* $3,864,270 in this example) is greater than or equal to the MinPortfolioValue parameter *(i.e.* $1,000,000 in this example). In this example, the TotalPortfolioValue for BasketID A127 is greater than the MinPortfolioValue required. Therefore, Basket ID A127 met the criteria for a valid program trade.

**[0063]** The process then proceeds to steps 172 and 174 where it zeroes out the PortfolioSymbolCount and the TotalPortfolioValue parameters. At step 176, the process sets the basket-in-process parameter to the latest basket identification number that was retrieved (*i.e.* BasketInProcess = B1743A). The process then returns to step 154 where it begins the process of analyzing BasketID B1743A.

[0064] The process continues to analyze and report the value of all the baskets that had crosses executed in the portfolio crossing session. When the process determines there are no more crosses to analyze at step 160, the portfolio crossing surveillance reporting is complete, as indicated at step 178.

[0065] While the invention has been discussed in terms of certain embodiments, it should be appreciated that the invention is not so limited. The embodiments are explained herein by way of example, and there are numerous modifications, variations and other embodiments that may be employed that would still be within the scope of the present invention.

## CLAIMS

1. A method performed at least partially on a computing system, the method comprising:

providing a crossing market center and designating a portfolio crossing session start time outside of normal market trading hours;

throughout the trading day until the designated portfolio crossing session start time, receiving and storing a plurality of program trade orders at the crossing market center using the computing system, wherein each program trade order represents a self-contained cross, contains a basket identification number identifying one program trade of a plurality of program trades on the crossing market center that the program trade order is part of, and contains a specified order price;

at the designated start time, initiating a portfolio crossing session using the computing system;

executing the received program trade orders at their specified order prices using the computing system, resulting in portfolio crosses;

linking together the completed portfolio crosses having the same basket identification number using the computing system; and

determining whether to validate each program trade by identifying a specific basket identification number in the linked

completed portfolio crosses for each program trade using the computing system, wherein validation occurs if a portfolio symbol count for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio symbol count for the crossing market center and a total portfolio value for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio value for the cross market center.

2. The method of claim 1, further comprising:

allowing the received program trade orders to be modified or canceled prior to the initiation of the portfolio crossing session.

3. The method of claim 1, further comprising

reporting the completed portfolio crosses to the appropriate Tape Reporting authority according to the rules of the Tape Reporting authority using the computing system.

4. The method of claim 3, wherein the rules of the Tape Reporting authority require the completed portfolio crosses to be reported as individual crosses.

5. The method of claim 3, wherein the rules of the Tape Reporting authority require the completed portfolio crosses to be reported in the aggregate.

6. The method of claim 1, further comprising

reporting the completed portfolio crosses to the trader that sent the initiating program trade order

using the computing system.

7. The method of claim 1, wherein the received and stored program trade orders are not disclosed to the marketplace.

8. The method of claim 1, further comprising reporting any set of portfolio crosses for a basket identification number that did not satisfy the criteria for a valid program trade using the computing system.

9. A method performed at least partially on a computing system, the method comprising:

providing a crossing market center and designating a portfolio crossing session start time outside of normal market trading hours;

throughout the trading day until the designated portfolio crossing session start time, receiving and storing a plurality of program trade orders at the crossing market center using the computing system, wherein each program trade order represents a self-contained cross, contains a basket identification number identifying one program trade of a plurality of program trades on the crossing market center that the program trade order is part of, and contains a specified order price;

at the designated start time, initiating a portfolio crossing session using the computing system;

executing the received program trade orders at their specified order prices using the computing system, resulting in portfolio crosses;

linking together the completed portfolio crosses having the same basket identification number using the computing system;

reporting the completed portfolio crosses to the appropriate Tape Reporting authority according to the rules of the Tape Reporting authority using the computing system; and

determining whether to validate each program trade by identifying a specific basket identification number in the linked completed portfolio crosses for each program trade using the computing system, wherein validation occurs if a portfolio symbol count for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio symbol count for the crossing market center and a total portfolio value for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio value for the cross market center.

10. The method of claim 9, further comprising:

allowing the received program trade orders to be modified or canceled prior to the initiation of the portfolio crossing session.

11. The method of claim 9, wherein the rules of the Tape Reporting authority require the completed portfolio crosses to be reported as individual crosses.

12. The method of claim 9, wherein the rules of the Tape

Reporting authority require the completed portfolio crosses to be reported in the aggregate.

13. The method of claim 9, further comprising reporting the completed portfolio crosses to the trader that sent the initiating program trade order using the computing system.

14. The method of claim 9, wherein the received and stored program trade orders are not disclosed to the marketplace.

15. The method of claim 9, further comprising reporting any set of portfolio crosses for a basket identification number that did not satisfy the criteria for a valid program trade using the computing system.

16. A crossing market center for processing program trades, comprising:

an interface for receiving program trade orders that constitute a plurality of program trades;

at least one crossing market center memory for storing code for analyzing and processing program trade orders and, having stored thereon, a portfolio crossing session start time outside of normal market trading hours;

at least one processor for interacting with the interface and executing the stored code for analyzing and processing program trade orders when the interface receives a program trade order, wherein each program trade order represents a self-contained cross, contains a basket

identification number identifying one program trade of the plurality of program trades on the crossing market center that the program trade order is part of, and contains a specified order price;

wherein the code, when executed, stores the program trade order and at the designated start time, initiates a portfolio crossing session;

executes the received program trade orders at their specified order prices, resulting in portfolio crosses;

links together the completed portfolio crosses having the same basket identification number; and

determines whether to validate each program trade by identifying a specific basket identification in the linked completed portfolio crosses for each program trade, wherein validation occurs if a portfolio symbol count for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio symbol count for the crossing market center and a total portfolio value for the linked completed portfolio crosses having the specific basket identification number is greater than or equal to a minimum portfolio value for the cross market center.

17. The method of claim 1, wherein the computing system is one or more programmed computers.

18. The method of claim 9, wherein the computing system is one or more programmed computers.

19. The method of claim 1, wherein the computing system is distributed over several physical locations.

20. The method of claim 9, wherein the computing system is distributed over several physical locations.

21. The crossing market center of claim 16, wherein the crossing market center is a computing system.

22. The method of claim 1, wherein the minimum portfolio symbol count is 15 and the minimum portfolio value is $1,000,000.

23. The method of claim 1, wherein the minimum portfolio symbol count is 15 and the minimum portfolio value is $1,000,000.

24. The crossing market center of claim 16, wherein the minimum portfolio symbol count is 15 and the minimum portfolio value is $1,000,000.