Algorithmic Trading Accelerator Guide

Version 9.10

April 2016

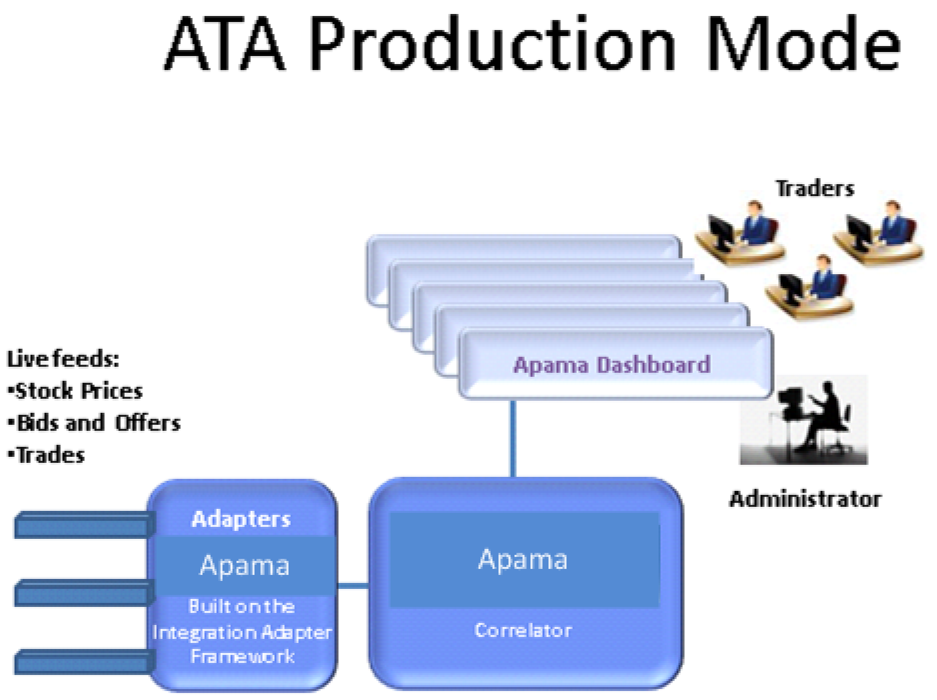
Document ID: PAF-ATA-910-20160415

Table of Contents

1 Overview of the Algorithmic Trading Accelerator

The Algorithmic Trading Accelerator (ATA) installs with the Capital Markets Foundation (CMF). Unlike solutions that offer commoditized, pre-defined strategies, the ATA enables you to quickly develop, refine, and deploy unique algorithmic trading strategies built upon your own intellectual property. The ATA relies on Apama's patented Event Stream Processing (ESP) platform to apply strategies against high volume market activity, enabling users to achieve trading objectives.

Apama includes an extensible framework for integrating with market data feeds, middleware technologies, order management systems and databases. And it offers a rich dashboard environment for graphical monitoring of algorithmic trading execution. The following illustrates the runtime architecture of a solution based on the ATA.



In a production environment, adapters feed events to an Apama correlator, which filters events and applies trading strategies. An administrator can set up risk firewalls to prevent unauthorized trading. Traders can view the orderbook and trade using the supplied Apama dashboard or a customized trading user interface. For an example of a custom trading interface, see the FX Aggregation solution, which is also based on the CMF.

ATA includes tools to monitor market flow and depth and to execute orders through one or more of the following:

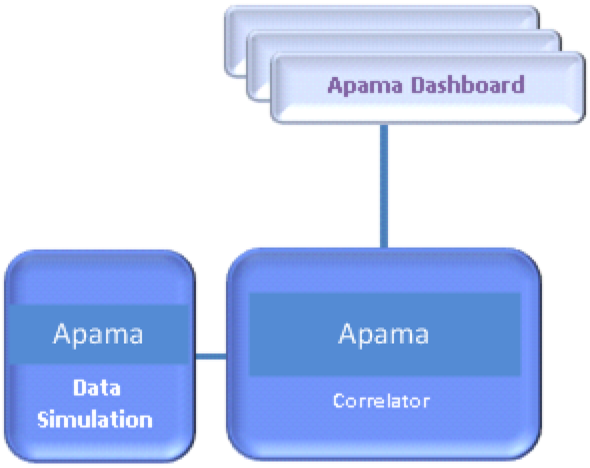
- Basic Direct Market Access (DMA) with explicit buy and sell orders.
- Execution distribution strategies such as Volume-weighted and Time-weighted Average Price (VWAP and TWAP), order slicing, iceberg, and Percentage of Volume (PoV).
- Market-responsive analytic strategies such as momentum trading, and crossover.
- Complex algorithms such as statistical arbitrage that are sophisticated in terms of the logical constructs that define them as well as the technological requirements to run them effectively.

All order activities can be monitored by several overlapping trading firewalls that are defined for sets of users, venues, and instruments. Each firewall monitors prices, quantities, positions, and client balances to alert traders at one threshold, and reject orders at the outer limit. The firewalls protect the client's interests when order submission and trade execution are virtually immediate.

Using ATA as a starting point for a custom solution allows you to:

- Develop unique automated trading strategies in a graphical modeling environment.
- Package existing algorithmic modules for reuse in new strategies.
- Build upon a foundation library of trading algorithms.
- Evaluate and refine strategies by testing against historical data.
- Execute thousands of trading scenarios concurrently.
- Use packaged connectivity for access to financial feeds.
- Optimize trade opportunities for both sell-side and buy-side firms.

Out-of-the-box, ATA provides a trade simulator and start-up scripts, which make it easy to explore ATA and learn about its capabilities. The following illustration shows the demonstration architecture.



This document shows you how to set up the Algorithmic Trading Accelerator, run it in demonstration mode, explore its client screens, set up distributed clients, and use Software AG Designer to create custom trading algorithms.

2 Running the ATA samples

The ATA includes algorithmic trading strategies that are injected into an Apama correlator, and an Apama dashboard that supports trading, displays information about the strategies, and allows you to configure firewall rules. Included scripts allow you to start the runtime pieces and demonstrate ATA capabilities.

You can run ATA on a single machine, or you can run the correlator on one machine and run multiple dashboards that connect to it remotely. You might do this for example, if you wanted to install the correlator on a machine running Linux. The dashboards must be run on Windows machines.

Note: Before you can run ATA, you must install and configure Apama and the CMF as described in *Capital Markets Foundation Guide*.

The CMF installation makes it easy to run the pre-defined ATA immediately on the local machine.

The ATA is copied into the %APAMA_WORK% directory during installation and run from there.

Then, in "Developing and deploying custom ATA applications" on page 31, you see how to import the projects into Software AG Designer, so you can define, test, and deploy your custom ATA.

The following table lists the scripts (located in ATA's installation directory %APAMA_WORK%/ATA_{version}) for starting and stopping the demo or the correlator and dashboard separately:

Description	Batch and sh file	Ant script
One command that: ■ Builds the project ■ Starts the correlator ■ Starts the trade simulator ■ Starts the Administrative dashboard	`startDemo`	■ `ant start-demo` ■ `ant start`
Starts the correlator.	`startServer`	`ant start-server`
Starts the Administrative dashboard.	`startClient` (`.bat` only, no equivalent on Linux)	`ant start-client`

Description	Batch and sh file	Ant script
Shuts down the correlator and trade simulator.	`stopDemo`	`ant stop`
Stops only the correlator.	`stopServer`	`ant stop-server`
Creates a deployment WAR which can be used to deploy the ATA using a Web Application Service. See "Deploying the ATA to a Web Application Server" on page 37 for usage information.	`deploy` (`.bat` only, no equivalent on Linux)	`ant deploy`
Prints a list of ant targets.	N/A	`ant usage`
Prints a list of ant targets.	N/A	`ant help`
Deletes log files and cleans up deployed strategies.	N/A	`ant clean`
Deletes log files.	N/A	`ant clean-logs`
Archives log files in a zip file in the `ATA` directory and deletes all log files from the `log` folder. The zip file suffix includes the date and time. For example, `log_20120420_103921.zip`. The correlators must be stopped before you run this command.	N/A	`ant archive-logs`

The following sections describe how to start and stop the ATA demo and walk through the available features and panels in the Algorithmic Trading dashboard. Later chapters describe how to set up users and customize ATA.

Starting and stopping the ATA demo

Demo scripts start a correlator, inject the ATA framework and strategies, start a market data simulator, and start an administrative dashboard--all on the same machine. A local ATA demo requires no authentication, and the user is referenced as your local login name, referenced here as a sample user myusername.

When your server session is done, stop the Algorithmic Trading accelerator server and perform a graceful shutdown. Closing the command prompt window where you started the server does not stop the process. You must explicitly run the stop command to stop the server.

Starting and stopping on Windows

On Windows, you can start and stop the ATA demo from the **Start > All Programs** menu, choose **Software AG > Tools > Apama Capital Markets *n.n* > Apama Capital Markets ATA *n.n* > Start Apama Capital Markets ATA *n.n*/ Stop Apama Capital Markets ATA *n.n***, or from a Apama Capital Markets command prompt, as follows:

1. To start and stop the ATA demo from the Apama Capital Markets command prompt:

    a. From the **Start > All Programs** menu, choose **Software AG > Tools > Apama Capital Markets Foundation *n.n* > Apama Capital Markets Command Prompt *n.n***.

    b. Change directories to `%APAMA_WORK%` directory and then to the top level ATA directory. For example:

    ```
    cd ATA_{version}
    ```

    c. Launch the demo start script by entering:

    `startDemo.bat` or `ant startDemo`

    The Algorithmic Trading dashboard **Home** page appears. The Home page provides access to the other pages as well as the status of the correlator and service.

    d. When you have completed your demo session, launch the demo stop script by entering:

    `stopDemo.bat` or `ant stopDemo`

Starting and stopping on UNIX or Linux

On UNIX or Linux, start and stop the ATA demo, as follows:

1. Open an Apama Command Prompt
2. Change directory to the root of the CMF installation.

3. Source `bin/CMF_env`
4. Enter `./startdemo.sh` or `ant startDemo`

   The Algorithmic Trading dashboard **Home** page opens. The Home page provides access to the other pages as well as the status of the correlator and service.
5. When you have completed your demo session, launch the demo stop script by entering `./stopdemo.sh` or `ant stopDemo`

Subscribing to Symbol Sets

Activities in Algorithmic Trading are based on market data. Whether running the demo and using simulated data or using real data from an adapter to an external feed, you first need to subscribe to the symbol sets of interest. Each symbol set corresponds to a data feed and initializes the symbol values for that feed.

To subscribe to a symbol set

1. From the dashboard **Home** page click **Setup** to open the **Symbol Set Subscription** page.
2. Select a symbol set.
3. Click **Subscribe**.

The demo simulates data from all these feeds, so you can subscribe to one or more. Each subscribed symbol set is highlighted and checked.

To unsubscribe from a symbol set, select it, and then click **Unsubscribe**.

Using the Order Management System

From the left navigation bar, click **OMS** to display the Order Management page. This page provides the following functionality:

- Selection of multiple instrument groups
- Visual summary of Bid/Ask/Last prices and volumes
- Visual summary of positions in each instrument
- Detailed view of full market depth (5 Best Bid/Asks)
- Order “blotter” view of all orders working and worked in the market
- Selection of an order provides detailed information, and enables modification or cancellation of live orders

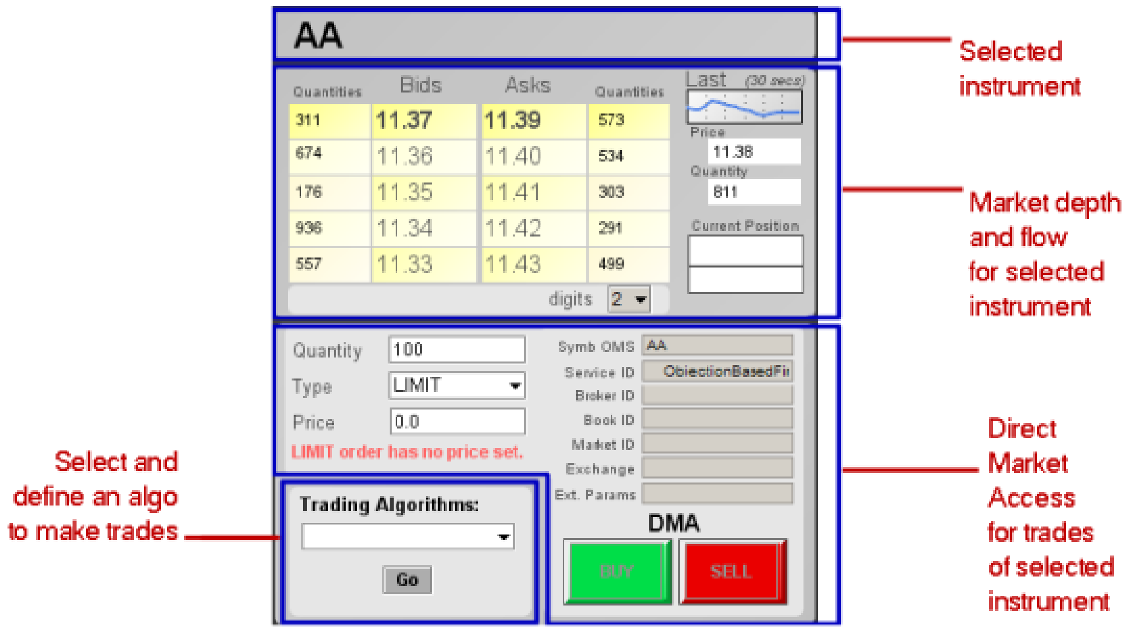
Note: The current position of all instruments will be persisted. When the application restarts, the positions are restored.

The Order Management System screen opens with the selected instruments of the subscribed symbol sets listed in the trader section.

At the top of the OMS screen, you can click a button to launch new panes of the Trader section (prices, market depth, and DMA) or the Blotter section (executed orders):

Trader section

The trader section provides the information about trades for the subscribed symbol set. The left side of Trader section provides a view of the market activity and holdings within the current instrument list. When you click on a line, it is selected for order activities. The right side of the Trader section enables monitoring of an instrument and access to trading of the selected instrument, as illustrated:


Selected instrument
Market depth and flow for selected instrument
Direct Market Access for trades of selected instrument
Select and define an algo to make trades
AA
Last (30 secs)
Current Position
digits 2
Trading Algorithms:
Go
DMA
BUY
SELL

Quantities	Bids	Asks	Quantities
311	11.37	11.39	573
674	11.36	11.40	534
176	11.35	11.41	303
936	11.34	11.42	291
557	11.33	11.43	499

Price 11.38
Quantity 811

Quantity 100
Type LIMIT
Price 0.0
LIMIT order has no price set.

Symb OMS AA
Service ID ObjectionBasedFi
Broker ID
Book ID
Market ID
Exchange
Ext. Params

The market depth and flow information take the pulse of the market for the selected instrument. You can then specify the quantity, type, and price of the intended trade. Clicking the **Buy** or **Sell** button submits the order, and then records the transaction in the blotter section. In addition to standard DMA capabilities, this screen supports launching trading strategies directly from this panel. For single instrument strategies, the dialog will reflect the selected instrument and its symbol set in the algo creation dialog.

You can open multiple Trader windows to monitor selected instruments.

Blotter section

The Blotter section provides the status of orders about trades for the subscribed symbol sets. It also will reflect orders that have been blocked by the firewall. You can clear the blotter listings; however, underlying open orders will continue to process. You can open multiple blotter windows but they will all reflect the same data and when one is cleared, they all are cleared.

Blotter Window Fields

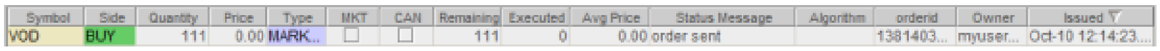
The Blotter window fields are:

Field Name	Description
Symbol	The name of the symbol
Side	The side of the symbol where BUY has a green background, and SELL has a red background.
Quantity	The total quantity requested by this order
Price	The price issued with this order
Type	The type of order: `MARKET`, `LIMIT`, `FOREX MARKET`, `FOREX LIMIT`, or `PREVIOUSLY QUOTED`
MKT	When checked, order is acknowledged to be in the market
CAN	When checked, order has been cancelled
Remaining	The quantity not yet traded on this order
Avg Price	The average price gained on this order
Status Message	The latest status message – click on order to see full list
Algorithm	OMS is from this dashboard, alternatively it will show the name of the algorithm that issued the order
Issued	The time and date that the order was issued

Background Effects

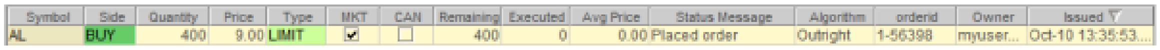
Backgrounds and text colors indicate an order's status:

Grey background. Order issued; not yet acknowledged from the market



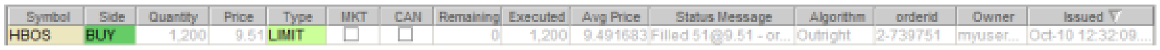
Symbol	Side	Quantity	Price	Type	MKT	CAN	Remaining	Executed	Avg Price	Status Message	Algorithm	orderid	Owner	Issued
VOD	BUY	111	0.00	MARK...	☐	☐	111	0	0.00	order sent		1381403...	myuser...	Oct-10 12:14:23...

Yellow background. Order acknowledged and working in the market



Symbol	Side	Quantity	Price	Type	MKT	CAN	Remaining	Executed	Avg Price	Status Message	Algorithm	orderid	Owner	Issued
AL	BUY	400	9.00	LIMIT	☑	☐	400	0	0.00	Placed order	Outright	1-56398	myuser...	Oct-10 13:35:53...

White background. Order acknowledged and no longer in market (completed, cancelled, rejected, etc.)



Symbol	Side	Quantity	Price	Type	MKT	CAN	Remaining	Executed	Avg Price	Status Message	Algorithm	orderid	Owner	Issued
HBOS	BUY	1,200	9.51	LIMIT	☐	☐	0	1,200	9.491683	Filled 51@9.51 - or...	Outright	2-739751	myuser...	Oct-10 12:32:09...

Order Details

Clicking on an order line displays details about the selected order.

In this panel you can:

- Modify the quantity, price, and type of the order (if the underlying exchange supports these changes).
- Cancel the order
- View history of all activity recorded on that order.
- When an order is complete, its history and the details are displayed.

A completed order can no longer be edited or cancelled.

Using Algorithmic Trading Strategies

The Algorithmic Trading Accelerator provides several pre-defined strategies that were created in the Apama Scenario Manager.

Note: You will learn later how to modify them to derive your own algorithms, and to redefine the dashboards in the Apama Dashboard Studio to suit your needs.

Algos Home Page

The **Algorithmic Strategies** page's **Home** tab lists and describes all the available strategies.

To select a strategy, either choose it on the pull-down menu, or click on the strategy in the scrolling list.

Note: The strategies provided in the installation are intended for demonstration and might be used as starting points for custom strategies. These strategies are not intended for production use until you have validated their behaviors in thorough offline tests with your adapters under real market conditions.

Crossover

The Crossover scenario calculates two moving averages for a given instrument—one over a long period and the other over a shorter period. The length of the windows can be configured to suit market and instrument volatility. A trading opportunity is identified when the two moving averages cross each other. The direction of the cross directs the side of the order. When the long term line rises above the short term line, a BUY order is issued for the clip quantity. When the short term line rises above the long term line, a SELL order is issued for the clip quantity.

Select an instance in the Scenario Instances to view its data in the graphic section.

To run strategies, click **Create New**. In the **Crossover: Create** dialog box, choose from the symbolsets to which you are subscribed, and then an instrument in the selected set.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want. This type of strategy has no defined end, so you need to explicitly stop each strategy instance.

This strategy exposes an important concept. The strategies that you create in a dashboard are active in the server on behalf of the current user. When you close all your running dashboards, the strategy will continue until the server is stopped. When you reconnect later from any browser, the active strategies can be viewed. After a server restart, the strategies and pending orders are not active. The user's position in each instrument was persisted and is displayed in the dashboard.

Iceberg

The iceberg strategy (also known as wave-trading, clipping, and quantity hiding) splits a given quantity into a number of smaller volumes, and then regularly issues partial orders into the market for the full order. This allows potentially large orders to be filled with minimum market impact and helps with situations of poor liquidity. The dashboard shows the progress of each scenario in summary and in detail when selected.

Select an instance in the Scenario Instances to view its data in the graphic section.

To run strategies, click **Create New**. In the **Iceberg Trading: Create** dialog box, choose from the symbolsets to which you are subscribed, and then an instrument in the selected set.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Momentum Trading

Momentum trading identifies a strong trend in the market and attempts to follow that trend. When a trend is identified it takes up a position, and then monitors the market. If the trend continues, it takes up another position, and continues to do so. Once the trend is identified as completed, the position built up is then neutralized. This scenario is intended for use on large trends in the market rather than constant intra-day trading such as statistical arbitrage.

Select an instance in the Scenario Instances to view its data in the graphic section.

To run strategies, click **Create New**. In the **Momentum Trading: Create** dialog box, choose from the symbolsets to which you are subscribed, and then an instrument in the selected set.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Order Slicing

Order slicing is a strategy that uses VWAP over a time window to trigger placing orders in the market. Order execution is wave traded as a limit order until a timeout occurs. It is then converted to market order for any remaining shares.

Note: To view data in the upper graph, you must first select an instance in the table.

You can choose to see **Average Price** instead of **Remaining**:

To run strategies, click **Create New**. In the **Order Slicing: Create** dialog box, choose from the symbolsets to which you are subscribed, and then choose an instrument in the selected set.

Parameters for order slicing include:

- Duration (strategy run time)
- VWAP Period (time window to calculate the vwap value)
- Max time in market (time period in market as limit order before any remainder is converted to a market order).
- Limit Offset (time limit offset (price additive at which to sell)
- Limit time (time in market before re-price against new vwap)

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Percentage of Volume (POV)

The Percentage of Volume execution strategy follows the volume traded in the market and distributes a specified quantity over time, in proportion to the total quantity traded in the market. This is much like iceberg trading an order but will follow market volume rather than trading clips of a fixed size. The purpose of this algo is to minimize market impact while ensuring that the desired quantity is traded. The main bar chart shows the volume traded in each clip so far.

Select an instance in the Scenario Instances to view its data in the graphic section.

The total order quantity is partitioned over time intervals, with child order sizes determined by the volume seen traded in market over the preceding interval. It is possible to cap the child order size if you want; otherwise, the algorithm will attempt to place orders of the specified percentage of market volume. Each child order will be worked in market, becoming more aggressively priced if it remains unfilled, and the aggression of this working is determined by the algorithm’s progress relative to the market volume.

Using the POV Strategy

The strategy relies on the average daily volume (ADV) being defined for each instrument used. To load ADVs into memory, click **Load ADVs**. The **POV Data Loader** dialog box appears.

You can load ADVs from a file. It is not essential to load ADVs, but if this step is skipped the user must specify the ADV for each strategy instance.(A sample file is provided to demonstrate the required format, but note that the sample values are not suitable for trading.)

To create a new strategy, click **Create New Strategy Instance**. The **POV Order Creator** dialog box appears.

Specify the following parameters:

Parameter	Value
Symbolset	The parent symbolset of the chosen symbol
Instrument	The symbol to trade
Total Quantity	The total quantity of the POV order to be placed
Percentage to Trade	The percentage of the total quantity chosen to actually send to the market.

Parameter	Value
Child Order Interval	The duration of each clip
Randomness	Random variation in seconds added to the child order interval.
Target % of ADV	The level of participation in the market; the strategy will not exceed this percentage of market volume.
Round Lot Size	The size of a round lot in the market; the strategy will quantise child orders at this granularity.
Max Child Order Size	(Optional) the maximum quantity to trade in a clip.
Max aggression	The aggression level at which to cap the used pricing models.
Side	**BUY** or **SELL**
Order Type	Limit orders will not trade worse than the specified price, and will not work child orders up to market orders. As with any limit order, complete execution is not guaranteed.
Limit Price	Only applied to limit orders. Used to cap/floor child order prices, and to restrict which trades in market are counted for determining eligible volume.
Max eligible % of ADV	Trades over this percentage of the ADV will be excluded from the market quantity when determining the size of child orders. This is intended to remove from consideration trades that are large enough to distort the market, and the resultant effect that the strategy would place large child orders in the wake of such trades.
ADV	If set to 0, the system will use a pre-loaded ADV. If this is set to 0 and no ADV is loaded, the strategy will not trade and must be manually deleted after creation.
Start Time	Start hour/minute of the strategy, which should fall within the start and end time of the chosen volume curve

Parameter	Value
End Time	End hour/minute of the strategy, which should be after the start hour/minute and fall within the start and end time of the chosen volume curve.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Statistical Arbitrage

This is a Bollinger-Band based statistical arbitrage trading scenario. It takes a pair of instruments and calculates the spread between their prices. From this spread, the Bollinger bands are calculated. Bollinger bands use the Standard Deviation calculation to build a pricing corridor around the spread price. When the spread crosses one of the bands, the strategy has identified a trading opportunity. Depending on which boundary is crossed: upper or lower, the orders are to BUY/SELL or SELL/BUY. You always buy one instrument and sell the other, and ideally have the same number of BUYs and SELLs, so you end up with no inventory.

Select an instance in the Scenario Instances to view its data in the graphic section.

To run strategies, click **Create New**. In the **Statistical Arbitrage: Create** dialog box, choose from the symbolsets to which you are subscribed, and then two instruments in the selected set.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Time Weighted Average Price (TWAP)

This strategy will trade a given volume over a given duration. The mean rate of trading is constant over time.

The total quantity to trade is partitioned into a number of clips; each clip is executed in turn throughout the trading period. The strategy will randomize clip sizes, and will work each order aggressively in market. The aggressiveness level is determined by discrete thresholds according to how far ahead or behind the target progress the TWAP is currently performing. The strategy will switch price models to become more or less aggressive in the market the farther ahead or more behind it is. It will place a market order as a last resort if trailing significantly. The main chart shows the volume traded in each clip.

Select an instance in the Scenario Instances to view its data in the graphic section.

Using the TWAP Strategy

The strategy relies on the average daily volume (ADV) being defined for each instrument used. To load ADVs into memory, click **Load ADVs**. The **TWAP Data Loader** dialog box appears. You can load ADVs from a file. It is not essential to load ADVs, but if this step is skipped the user must specify the ADV for each strategy instance.(A sample file is provided to demonstrate the required format, but note that the sample values are not suitable for trading.)

To create a new strategy, click **Create New Strategy Instance**. The **TWAP Order Creator** dialog box appears.

You can choose to limit total trades to a specified percentage of the average daily volume. To do so, the ADV must first be loaded from a CSV file via the Load ADVs dialog. Once this is done, or if volume limiting is not to be used, click the Create New Strategy Instance button to launch a strategy creation dialog, and then enter values for the TWAP parameters:

Parameter	Value
Symbol Set	The parent symbolset of the chosen symbol.
Instrument	The symbol to trade.
Total Quantity	The total quantity of the TWAP order to be placed.
Slice Interval Size	The interval size for slices in units of minutes.
Subslicing Frequency	The interval size for subslices in units of seconds (not a true frequency; nomenclature derives from a specification document).
Side	BUY or SELL.
Volume Curve Name	The name of the volume curve to be used for determining the slice quantities Child Order Interval.
Round Lot Size	The quantity by which orders and the total quantity should be rounded.
Max Aggression	The aggression level at which to cap the used pricing models.

Parameter	Value
Percent to Trade	The percentage of the total quantity chosen to actually send to the market.
Max % of ADV	The percentage of the average daily volume (ADV) at which to cap the total quantity. (Optional, defaults to 100%)
Randomness	Random variation in seconds added to the subslicing frequency.
Start Time	Start hour/minute of the strategy, which should fall within the start and end time of the chosen volume curve.
End Time	End hour/minute of the strategy, which should be after the start hour/minute and fall within the start and end time of the chosen volume curve.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Volume Weighted Average Price (VWAP)

Also referred to as Volume Curve based trading, this strategy will trade a given volume over a given duration with respect to a defined trading volume.

Note that the volume curve will be displayed over its full length, typically a trading day, while the planned/executed volumes will be displayed only over the strategy lifetime.

Select an instance in the Scenario Instances to view its data in the graphic section. The user provides the predicted volume distribution curve in either absolute values or in percentages. This defines what the trader believes the trading volume will be over the duration of that time window.

The total quantity to trade is then partitioned into a number of clips; each clip is executed in turn throughout the trading period. In addition to calculating the correct volume to execute at each clip, the scenario will work each order aggressively. The aggressiveness level is determined by discrete percentage thresholds according to how far ahead or behind the anticipated market volume the VWAP is currently performing. The strategy will switch price models to become more or less aggressive in the market the farther ahead or more behind it is. It will place a market order as a last resort if trailing significantly.

Using the VWAP Strategy

The strategy relies on the average daily volume (ADV) being defined for each instrument used. To use this strategy, click **Load Volume Curves/ADVs**. The **VWAP Data Loader** dialog box appears. You can load ADVs from a file. It is not essential to load ADVs, but if this step is skipped the user must specify the ADV for each strategy instance.(A sample file is provided to demonstrate the required format, but note that the sample values are not suitable for trading.)

After volume curves (and ADVs, if specified) are loaded, click **Create New Strategy Instance** to open a strategy creation dialog. The **VWAP Order Creator** dialog box appears.

In the dialog box, specify values for the following VWAP parameters:

Parameter	Value
Symbol Set	The parent symbolset of the chosen symbol.
Instrument	The symbol to trade.
Total Quantity	The total quantity of the TWAP order to be placed.
Slice Interval Size	The interval size for slices in units of minutes.
Subslicing Frequency	The interval size for subslices in units of seconds (not a true frequency; nomenclature derives from a specification document).
Side	BUY or SELL.
Volume Curve Name	The name of the volume curve to be used for determining the slice quantities Child Order Interval.
Round Lot Size	The quantity by which orders and the total quantity should be rounded.
Max Aggression	The aggression level at which to cap the used pricing models.
Percent to Trade	The percentage of the total quantity chosen to actually send to the market.
Max % of ADV	The percentage of the average daily volume (ADV) at which to cap the total quantity. (Optional, defaults to 100%)

Parameter	Value
Randomness	Random variation in seconds added to the subslicing frequency.
Start Time	Start hour/minute of the strategy, which should fall within the start and end time of the chosen volume curve.
End Time	End hour/minute of the strategy, which should be after the start hour/minute and fall within the start and end time of the chosen volume curve.
Volume Curve Name	The name of the volume curve to be used for determining the slice quantities Child Order Interval.

When you change the value of a parameter, press Enter (while focus is on that field) to persist the change.

Click **Create** to save the parameters, and then click on its line on the page to run the strategy. Create as many strategies as you want.

Using Risk Firewalls

Financial institutions and investors need to prevent unacceptable trading practices from causing damage. Many traders have a large appetite for risk, so they willingly push the envelope. In doing so, a series of trades can violate house or client limits. The firewall rulesets in this chapter show how administrators can define rules that track all the trading activities on the order desk, recording the specific limits and violations of rules. The result is a comprehensive picture of risk mitigation. The **Home** page of the risk firewalls provides access to the overview of rule violations as well as each of the ruleset pages.

Note: Instances of firewalls will be persisted. When the application restarts, the firewalls are restored.

Overview of Active Risk Firewalls

The **Overview** page in the Firewalls section provides a monitoring station to review the rules that are in effect currently, any outstanding alerts, and details on a selected alert. The **Lock/Unlock Firewall** button allows the Risk Firewall to be locked, which means that it rejects all orders without querying the rules.

Viewing Firewall Rule Details

Each rule can be evaluated on **Rules Detail** page, showing all instances of all rules.

Firewall Ruleset: Order Price Limits

Order Price Limit rules define at what price range specified traders can submit orders for specific instruments before an alert or an objection is raised. The top section of the page is where new rules are defined. Clicking **Add** enters the rule in the **Instances** list. You can modify an order-price-limit rule by selecting a rule, and then clicking **Modify**. A dialog appears displaying the **Warning/Objection** settings.

Note: You cannot modify the list of instrument symbols, or trader identifiers in a rule instance. If you need to do that, you must delete the existing instance and then create a new one.

After making changes, click **Modify** to update the rule.

Firewall Ruleset: Order Quantity Limits

Order Quantity Limit rules define at what volume range specified traders can submit orders for specific instruments before an alert or an objection is raised.

The top section of the page is where new rules are defined. Clicking **Add** enters the rule in the **Instances** list.

You can modify an order-quantity-limit rule by selecting a rule, and then clicking **Modify**. A dialog appears displaying the **Warning/Objection** settings.

Note: You cannot modify the list of instrument symbols or trader identifiers in a rule instance. If you need to do that, you must delete the existing instance and then create a new one.

After making changes, click **Modify** to update the rule.

Firewall Ruleset: Position Limits

Position Limit rules define the maximum position that can be held in an instrument for all traders before an alert or an objection is raised.

The top section of the page is where new rules are defined. Clicking **Add** enters the rule in the **Instances** list.

You can modify an order-price-limit rule by selecting a rule, and then clicking **Modify**. A dialog appears displaying the **Warning/Objection** settings.

Note: You cannot modify the list of instrument symbols or trader identifiers in a rule instance. If you need to do that, you must delete the existing instance and then create a new one.

After making changes, click **Modify** to update the rule.

Firewall Ruleset: Order Throttle Limits

The risk firewall rules for **Order Throttle Limit** define the rate at which specified traders can submit orders for specific instruments before an alert or an objection is raised.

The top section of the page is where new rules are defined. Clicking **Add** enters the rule in the **Instances** list.

You can modify an order-throttle-limit rule by selecting a rule, and then clicking **Modify**. A dialog appears displaying the **Warning/Objection** settings.

Note: You cannot modify the list of instrument symbols or trader identifiers in a rule instance. If you need to do that, you must delete the existing instance and then create a new one.

After making changes, click **Modify** to update the rule.

Firewall Ruleset: Client Credit Limits

Client Credit Limit rules define how a customer's account value can be impacted before an alert or an objection is raised. This Firewall rule uses the price of the order being placed to determine the order value. When placing a Market order, the price in the order will generally be zero as it is ignored. This means that you could breach the firewall warning/objection limits with that order. Once the limits have been breached, you will only be allowed to place orders that bring the account value back into acceptable limits. The top section of the page is where new rules are defined. Clicking **Add** enters the rule in the **Instances** list.

Using Market Test Scenarios

Embedded within the Algorithmic Trading Accelerator and under the “Market Tests” table in the dashboard are two low-level testing scenarios – one for Market Depth and one for Order management. These are orthogonal to the rest of the Algorithmic Trading Accelerator in that they do not utilize symbol sets. The fields entered are mapped directly to the input variables on the CMF market data and order management SmartBlocks.

Note: Because the test scenarios allow a user to bypass any configured firewall rules and send orders directly to a connected adapter, these scenarios will typically be removed from the production deployment of an Algorithmic Trading solution.

These scenarios are useful for testing integration of Apama with market data and order management adapters.

Testing with the Order Manager Scenarios

The Order Manager test scenario allows a user to submit and manage orders directly to a connected adapter or exchange simulator, optionally bypassing any trading restrictions configured by the firewall. The order parameters on the dashboard are mapped to the corresponding input variables on the standard Order Manager smart block. Refer to the documentation for these smart blocks for more details of how their parameters are interpreted.

To submit a new order, fill in the required parameters, and then click **Create**. Usually, the following parameters need to be specified:

- Symbol
- Price
- Quantity
- Side
- Type
- ServiceId
- Market

The service id and market values are those used to configure the adapter session where the order will be submitted. Consult the adapter documentation for more details of the service and market identifiers used by the specific adapter. Generally, these values are the same as those used in a symbol set that operates with that adapter.

Extra parameters can be passed to the order through the `ExtraParams` field in the form `name1=value1;name2=value2;...`

To submit an order through the risk firewall rather than directly to the adapter, set the `ServiceId` parameter to `__ObjectionBasedFirewallController` (notice that there are two leading underscores) and add `Firewall.TargetService=serviceId` to the `ExtraParams` field, where `serviceId` is the service identifier of the adapter to which the order should be sent after it has passed through the firewall (provided that it is not rejected by any of the configured firewall rules.)

Once an order has been submitted, it displays in the table at the bottom of the dashboard. When an order from this table is selected, its details display in the top part of the dashboard. The fields on the right hand side of the dashboard show details of any fills (executions) for the selected order.

The selected order can be cancelled by clicking **Cancel**, or amended by updating any parameters to be changed, and then clicking **Modify**. Clicking **Delete** deletes the scenario instance associated with the selected order, after attempting to cancel the underlying order. Clicking **Delete All** deletes all scenario instances, attempting to cancel any currently active related orders.

Testing with the Market Data Scenarios

The Market Data test scenario allows a user to directly subscribe to market data from a connected adapter or exchange simulator. The parameters on the dashboard are mapped to the corresponding input variables on the standard Market Depth and Market Depth and Order Flow smart blocks. Refer to the documentation for these smart blocks for more details of how their parameters are interpreted.

To subscribe to market data, fill in the required parameters, and then click **Create**. Usually, the following parameters need to be specified:

- Instrument
- ServiceId
- Market

Whether to subscribe to Market Depth and/or Tick (trade) data can be configured using the checkboxes above the Create button.

The service id and market values are those used to configure the adapter session from which the market data comes. Consult the adapter documentation for more details of the service and market identifiers used by the specific adapter. Generally, these values are the same as those used in a symbol set that operates with the that adapter.

Extra parameters can be passed to the order through the `ExtraParams` field in the form `name1=value1;name2=value2;...`

To submit an order through the risk firewall rather than directly to the adapter, set the `ServiceId` parameter to `__ObjectionBasedFirewallController` (notice that there are two leading underscores!) and add `Firewall.TargetService=serviceId` to the `ExtraParams` field, where `serviceId` is the service identifier of the adapter to which the order should be sent after it has passed through the firewall (provided that it is not rejected by any of the configured firewall rules.)

Once a subscription has been made, it will appear in the **Current subscriptions** table in the center of the dashboard. When a subscription from this table is selected, its details display in the top part of the dashboard. Any market data received from the subscription is displayed in the fields at the bottom of the dashboard.

The selected subscription can be cancelled by clicking **Delete**, or amended by updating any parameters to be changed, and then clicking **Commit**. Deleting a subscription also deletes the corresponding scenario instance. Click **Delete All** to delete all these scenario instances and cancel all the active subscriptions.

3 Developing and deploying custom ATA applications

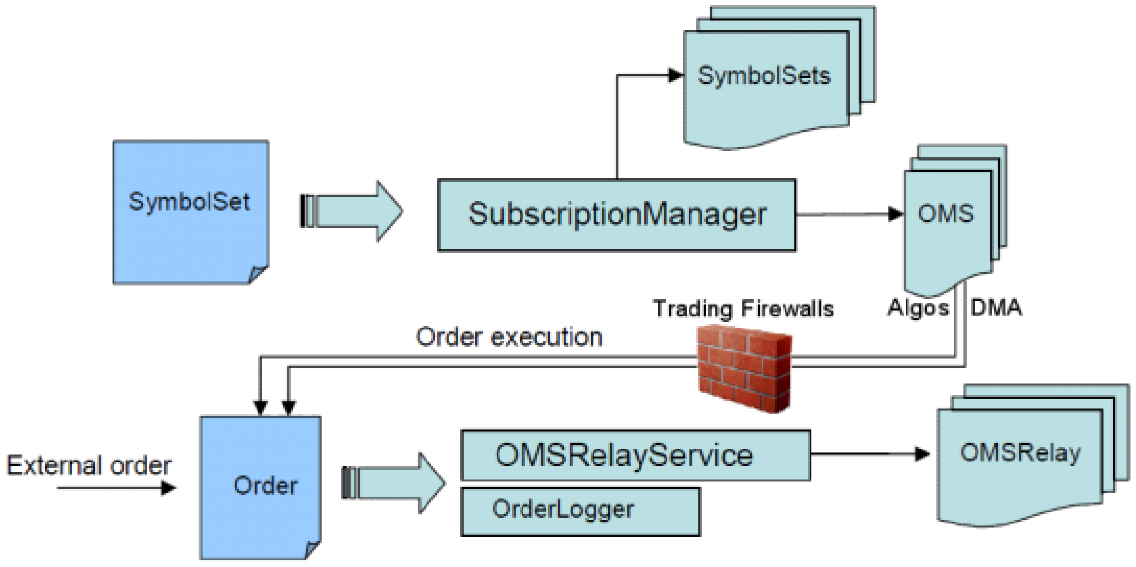
Once you have explored the pre-defined ATA locally, you are ready to start working on development of your custom ATA in Software AG Designer.

Architecture overview

In preparation for using the ATA to create a custom solution, you should be familiar with the architecture of its components and the hierarchy of its folders.

Components of the Algorithmic Trading solution

The Algorithmic Trading Accelerator contains the following components:



The functionality of an Algorithmic Trading Accelerator instance is determined by the sequence of events, monitors and scenarios injected into the Algorithmic Trading correlator when it is launched.

Directory structure

The following table describes the contents of the `Apama Capital Markets Foundation` *`n.n`*`/accelerators/ATA` folder:

Directory	Description
`adapters`	Location for IAF XML files (and any required libraries) that start adapters.
`build`	Events and monitorscript files that need to be injected into the Correlator are placed here when a

Directory	Description
	`ant build` command runs. Also, `WAR` deployment files are deployed from here.
`catalogs/ATA Blocks`	Includes utilities such as the `MarketParticipationGraph`. Events and monitorscript files that need to be injected into the Correlator are placed here when a `project build` command runs. Also, WAR deployment files are deployed from here.
`catalogs/ATA Functions`	include a function to extract values from an “extra params” field, specifically for extracting extra param values from those values associated with a Symbol Set.
`config`	Contains the dashboard configuration to be used when deploying dashboards.
`dashboards/algos`	Contains the sample algos. To add a new algo, create a new subdirectory, and then put your algo's dashboard files (RTV files) in that directory , together with image. files it uses. The parent directory (`dashboards`) also includes the dashboard project file and the framework directory (which includes the main Algorithmic Trading Accelerator dashboards). The only file in this folder that you should edit is `panels.ini`, which you should extend to make your custom Algos appear in the main dashboard panel. The background properties of your dashboards should be 850 wide by 650 high.
`extensions`	A folder of useful examples which can be used as illustrations or modified and used within the Algorithmic Trading Accelerator. Includes an example that dynamically creates SymbolSets.
`src`	Custom algos are defined in the six `src` directories. The folders enable complex scenarios to be constructed that require custom SmartBlocks (which should be stored in the `catalogs/ATA blocks` directory) and that require custom event types, service monitors and or initialization monitors and files.

Directory	Description
`src/Algo_Scenarios`	Create a folder for each of your Scenario Manager scenarios in a separate sub-folder.
`src/EventTypes`	Any custom event types (SmartBlocks or Service Monitors) required by your algo.
`logs`	The default location of log files for the Algorithmic Trading solution, the correlator, and any adapters started. Log file name patterns include: * `correlator_Log_`*`DATE`*`.log` – The correlator log file; also includes a log of all orders submitted and modified. *`DATE`* is time when the correlator was started. * `replay_Log_`*`DATE`*`.log` – The ReplayLog for the correlator that was started at *`DATE`*. * *IAF_File.xml.log* – A list of log files from the IAF process. Name of the log file is the name of the IAF XML file with a .log suffix.
`src/framework`	Contains the Scenarios and Monitors that make up the Algorithmic Trading Accelerator.
`src/InitialiseFiles`	Any Monitors or event files required by your Algos/ SmartBlocks
`src/Monitors`	Any custom monitors required by your algo.
`symbolsets`	A directory of SymbolSets event files

The `Apama Capital Markets Foundation` *`n.n`* folder also contains resources for ATA:

Directory	Description
`bin`	Script that sets up the CMF environment.
`doc`	Contains the menu, `start.htm`, which links to the PDF books that document the Capital Markets solutions. Also stages the help `JAR` that is installed in the Software AG Designer. Provides access to the online API reference when you open `doc\reference\index.html` in a browser.

Directory	Description
`lib`	All the external libraries required by the Algorithmic Trading solution.

Importing the ATA Project into Software AG Designer

When you have Software AG Designer and CMF installed, you can modify the ATA that is pre-defined as an Eclipse project.

Note: If you ran the ATA through its startDemo script, an ATA_{version}\build directory was created. That directory will cause issues if it is imported as part of an existing project. You must delete the ATA_{version}\build directory before starting the project import procedure. The project will then create its build directory in its workspace.

To import the Algorithmic Trading project

1. In Software AG Designer, choose the **Apama Developer** perspective.
2. Select **File > Import**.
3. On the **Import** dialog box, select **General > Existing Projects into Workspace**.
4. Click **Next**.
5. Confirm that the option **Copy projects into workspace** is selected.
6. At **Select root directory**, click **Browse**.
7. Select the **ATA** directory that you copied to your custom location, and then click **OK**.
8. Click on `Algorithmic Trading Accelerator`.
9. Click **Finish**.
10. Select **Project > Clean**.
11. Choose to rebuild all projects.

    The project is imported and built in the Software AG Designer.
12. You can run the ATA within the Software AG Designer by right-clicking on the `Algorithmic Trading Accelerator` project, and then choosing **Run as > Apama Application**.

    Note: Using this launch profile runs the Apama server and a dashboard locally. The launch expects to start the correlator, so check to see that it is stopped before performing a launch. (Before using the launch profile for the first time, you should close and re-open the Algorithmic Trading project, or restart Software AG Designer. That will work around an issue in Software

AG Designer where the block and function catalogs of a newly imported project might not be found on its first run.)

Quick steps to a new algorithm

Defining a new algorithm can be as easy as copying an existing one, and then adjusting its names and functions to make it your own. The following copy/paste/rename tasks can be performed in the file system at the corresponding locations in the `\Users\{username}\workspace99`.

To define a new algorithm from an existing one

1. On the Software AG Designer's **Project** menu, clear the option on **Build Automatically**.
2. Three elements are required to clone a strategy. The strategy implementation that has built in the Apama Event Modeler, the dashboards that display the strategy and its dialogs, and the initialization events that provide access to the strategy from relevant lists and pulldowns.

   If you encounter synch alerts, do a refresh (F5) to resynchronize the files.
3. To clone the `Crossover` strategy:
   a. Copy the folder `src/Algo_Scenarios/Crossover/` and then click on `src/Algo_Scenarios/` and paste it.
   b. Rename the folder `myCrossover`, and then rename the file `myCrossover.sdf`.
   c. Change something simple like the text message.
   d. Save the `myCrossover.sdf` file. This will now be injected when the correlator restarts.
4. The dashboards run and display the new strategy, and its dialog box for creating instances. To clone the dashboards
   a. Copy the folder `dashboards/algos/Crossover/`, and then paste it `dashboards/algos/`.
   b. Rename the folder `myCrossover`, and then rename the files `myCrossover.rtv` and `myCrossover_Create.rtv`
   c. Open each dashboard in Apama Dashboard Builder, revise the name label to `myCrossover`, revise the command's target scenario to `myCrossover`, and any other locations that are assigned the strategy name.)
   d. Save both dashboard files.
5. To add the new strategy to get it on the menu, copy the file `src/InitialiseFiles/scenario_metadata/CrossoverDescription.evt`Paste it as `myCrossoverDescription.evt`. Update the first two strings in the file to `"myCrossover"`.
6. Build and the run the project.

The algo menu displays your addition.

Choosing the **myCrossover** algo, and then clicking **Create** displays its dashboards.

Deploying the ATA to a Web Application Server

The ATA can run on a single machine, and allow multiple trading clients to connect to it using dashboards that have been deployed using a Web Application Server of the user's choice.

To deploy the ATA to a Web Application Server

1. If you need to adjust the dashboard deployment properties (such as the port that the display server uses), edit the file `%APAMA_FOUNDATION_HOME%/config/dashboard_deploy.xml` in the ATA directory to set any appropriate settings that are required for deployment.
2. In the **Start** menu, expand **Software AG > Tools > Apama Capital Markets *n.n*** and choose **Apama Capital Markets Command Prompt *n.n***. This sets up the environment of Apama and the CMF in a Command Prompt located in the working directory.
3. Change directories to set the path to the `ATA` directory, `%APAMA_FOUNDATION_HOME%/accelerators/ATA`.
4. Enter `deploy.bat`.

When successful, a WAR file for the deployment of the Algorithmic Trading Accelerator is created in the ATA's `deploy` directory. This WAR file can then be deployed as a Web Application Server of the user's choice. Users should follow the deployment instructions for their chosen Web Application Server, which should be run with the Java version supported by Apama.

This deployment WAR file will be signed with a self-signed certificate supplied with the ATA. This may cause authentication warnings on client machines with some Java runtime environment versions.

For a workaround to these warnings, see the following topic in the Apama documentation: Deploying and Managing Apama Applications > Deploying Dashboards > Applet and WebStart Deployment. A Certificate Signing Request (CSR) file, `DashboardKeystore.csr`, is included in the `ATA/dashboards` directory for this purpose. If you wish to sign the deployment WAR file with your own key store then update the properties in the `Deploy and Keystore properties` section of the `ATA/build.xml` file.

Adding Adapters

When you are ready to move away from test data to real data, you must access data feeds, and then set up appropriate Apama adapters that will deliver their data to the correlator.

1. Edit the file `adapters.xml` to define the adapters you want to start and reference their non-custom dependencies. You have access to the properties defined in the `APAMA_PROPERTIES` files to locate these: `${apama.home}` and `${apama.work}`.
2. Modify the ant xml script target adapters with the following inner tasks:
    a. <start-iaf>

    Starts an Adapter (.xml). The attributes you can use for the target are:

    - `config`: the xml adapter configuration file location.
    - `port`: the port to use for the iaf process.
    - `log`: the log file to use.
    - `loglevel`: the iaf log level to use.
    - `console`: when `true`, launch a separate console to launch the process.

    For example:

    ```
    <start-iaf config=”FIX.xml” loglevel=”ERROR” port=”16778” />
    <start-iaf config=”FIX.xml” />
    ```

    b. <engine-inject>

    Injects a MonitorScript file (.mon) located anywhere on disk. This also takes a <filelist> or <fileset> path definition. The attributes that can be used with this target are:

    - `host`: the target correlator host.
    - `port`: the target correlator port.
    - `file`: an optional file to send.
    - `parallel`: when true, all files to inject will be batched into one call.
    - `failonerror`: fails on the first engine send fault.

    For example:

    ```
    <engine-inject file=”my_file.mon” />
    ```

    c. <engine-send>

    Send event files (.evt). also takes a <filelist> or <fileset> path definition. The attributes that can be used with this target are:

    - `host`: the target correlator host.
    - `port`: the target correlator port.
    - `file`: an optional file to send.
    - `parallel`: when true, all files to inject will be batched into one call.
    - `failonerror`: fails on the first engine send fault.

    For example:

```
      <engine-inject file=”my_file.evt” />
```

For more information about specifying ant path types refer to the following resources:

- File lists: `http://ant.apache.org/manual/CoreTypes/filelist.html`
- File sets: `http://ant.apache.org/manual/CoreTypes/fileset.html`

The order of your entries defines the order in which they will injected or started. For example:

```
<target name="adapters" description="Main target for Apama adapters">
       <echo message="Starting adapters ..." />
       <engine-inject>
       <filelist>
            <file name=”file1.mon” />
            <file name=”file2.mon” />
       </filelist>
       </engine-inject>
       <engine-send>
       <filelist>
            <file name=”file1.evt” />
            <file name=”file2.evt” />
       </filelist>
       </engine-send>
       <start-iaf config=”my_adapter_config.xml” />
       <echo message="Adapters done." />
</target>
```

Adding Symbol Sets

The ATA infrastructure must be aware of the type of data that will be coming in on feeds. The Symbol Attributes smartblock handles this functionality. You can create a temporary symbol set in the dashboard to be used for demonstration or testing purposes or you can add a symbol set permanently by initializing it in the server.

Getting Symbol attributes

If you want to create a custom algorithm, you will use the Symbol Attributes SmartBlock that is distributed with the Algorithmic Trading Accelerator to handle symbols. The SmartBlock takes two parameters—the symbol name and the name of the symbol set to which the symbol belongs.

Given these names the SmartBlock requests the correct information from an underlying symbolset service and returns the required symbology details. The ATA supports two concurrent symbologies within a single SymbolSet. This allows one to be used to request market data and another to be used to trade upon (for example, receive Reuters RICs and trade a proprietary symbol via FIX.)

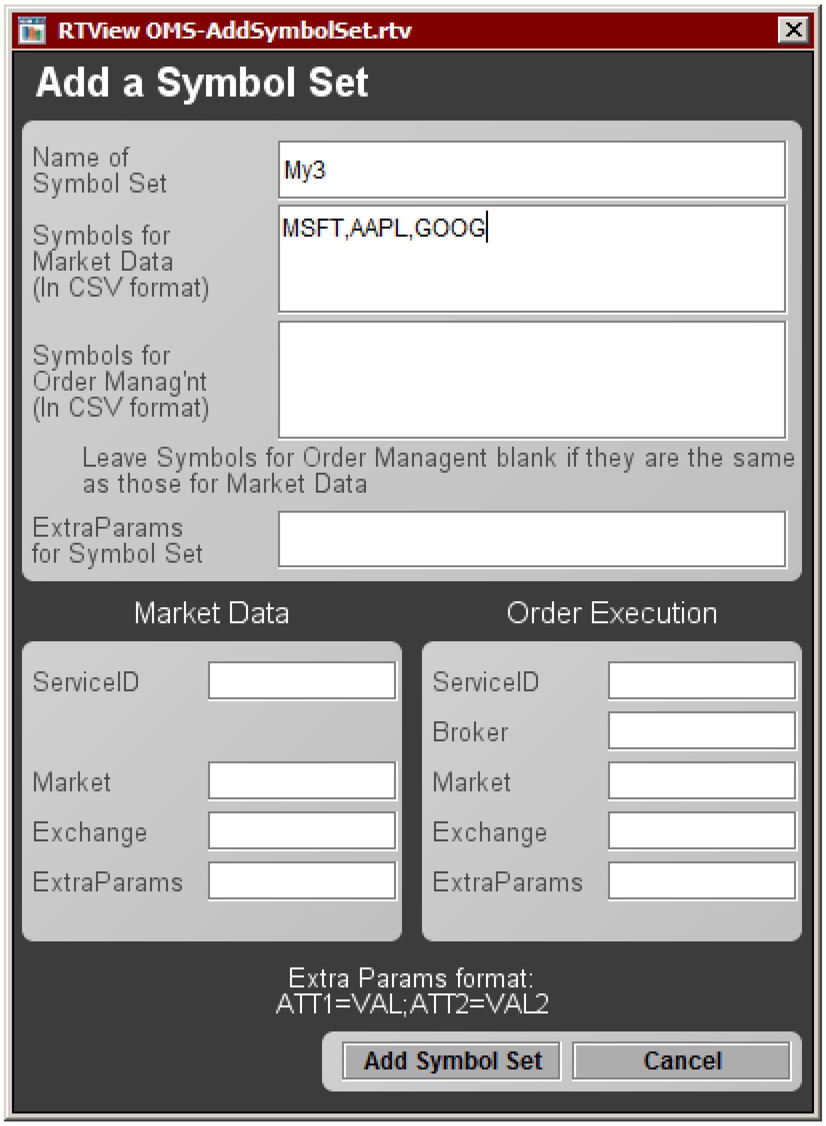
This information comes from three output feeds on the Symbol Attributes SmartBlock:

- Order Execution — This feed holds the following fields which are used to setup the Order Manager SmartBlock and in other places where the symbol details for order management (e.g. Order placement/amendment) are required.
  - Broker
  - Market
  - Exchange
  - extraParams
- Market Data — This feed holds the following fields which are used to setup the Market Depth SmartBlock and in other place where the symbol details for market data (e.g. Depth/Tick) are required.
  - Service Id
  - Broker
  - Market
  - Exchange
  - extraParams
- Symbols — The symbol for OMS and MarketData might vary on some platforms. Under such circumstance, the attributes block also returns the relevant symbol for OMS, remember that as input parameter it takes the symbol for marketdata.
  - SymbolForMarketData
  - SymbolForOMS

Adding temporary Symbol Sets

A client can create symbol sets that will be available to all users throughout the correlator session by using the **Add Symbol Set** dialog on their user display.

To create a temporary symbol set, click **Add Symbol Set** on the **Setup** page.


RTView OMS-AddSymbolSet.rtv

Add a Symbol Set

Name of Symbol Set	My3
Symbols for Market Data (In CSV format)	MSFT,AAPL,GOOG
Symbols for Order Manag'nt (In CSV format)	
ExtraParams for Symbol Set	

Leave Symbols for Order Managent blank if they are the same as those for Market Data

Market Data

ServiceID	
Market	
Exchange	
ExtraParams	

Order Execution

ServiceID	
Broker	
Market	
Exchange	
ExtraParams	

Extra Params format:
ATT1=VAL;ATT2=VAL2

Enter the values for your symbol set. In the illustration, the minimal data was entered: a name and a comma-delimited list of market symbols. Note that the entries are not quoted and the group is not set in brackets. Clicking **Add Symbol Set** injected the definition into the correlator, making it available to all users of dashboards connected to the server.

When the correlator is stopped, the temporary symbol sets are dropped, and are not reinstated when the correlator restarts.

Adding Symbol Sets on the server

When you define symbolset event files on the server in the `\Users\{username}\workspace99` folder, those symbolsets are available to all clients that login to that server, and persist through restarts of the server. You must define market and order symbols. This abstraction enables use of different adapters for markets and orders, allowing for variations in market symbol labels.

When you prepare to use actual adapters, each adapter will need to be configured as described in its documentation. The service, exchange, and market values of the adapter configuration will be values that you are propagating into a SymbolSet.

The following elements define a single event of the SymbolSet type:

```
event SymbolSet {
string id;  // the name of the symbol set, e.g. EUBonds
string serviceMKT; // the service ID to access prices/market data
string marketMKT; // the market ID under the service ID for market data
string exchangeMKT; // the exchange ID filter for market data
dictionary <string, string> extraParamsMKT; // Extra Params for market data
string serviceOMS;     // the service ID to access exchange to buy/sell
string brokerOMS;   // the broker ID under the service ID above
string marketOMS;         // the market ID under the service ID
string exchangeOMS;        // the exchange ID filter for the service ID above
dictionary <string, string> extraParamsOMS; // Extra Parameters for OMS
sequence<string> symbolsMKT;   // list of symbols for Flow/Depth
sequence<string> symbolsOMS;   // list of symbols for Orders
dictionary <string, string> extraParamsSymbolSet; // Extra Parameters for
               //  the SymbolSet
                }
```

For example, `SymbolSet-FX.evt` contains:

```
com.apama.ata.SymbolSet("FX","ADAPTERSERVICE","","",{},
   "ADAPTERSERVICE","","","",{},
["EUR/USD","USD/JPY","GBP/USD","USD/CHF","USD/CAD","AUD/USD",
   "NZD/USD","USD/SAR",
"USD/KWD","EUR/JPY","EUR/GBP","EUR/CAD","EUR/AUD","GBP/JPY",
   "EUR/CHF","GBP/CAD",
"CHF/JPY","AUD/JPY","AUD/CAD"],["EUR/USD","USD/JPY","GBP/USD",
   "USD/CHF","USD/CAD",
"AUD/USD","NZD/USD","USD/SAR","USD/KWD","EUR/JPY","EUR/GBP",
   "EUR/CAD","EUR/AUD",
"GBP/JPY","EUR/CHF","GBP/CAD","CHF/JPY","AUD/JPY","AUD/CAD"],
   {"SIMULATION_MODE":"DEFAULT"})
```

Note that the service, market and exchange values need to have position placeholders in the string, and that "" will defer each field to its default setting.

Consider another example, `SymbolSet-myEquities.evt`:

```
com.apama.ata.SymbolSet("myEquities","Reuters","EQUITIES","NASDAQ",
      {"ATT1":"VAL1", "ATT2":"VAL2"},"FIX","JPMC","EQUITIES","NASDAQ",
      {"OMStimeout":"10","OMSwindow":"OPEN"}, ["AAPL","MSFT","APMA"],
  ["AAPL","MSFT","APMA"], {})
```

Note: When you leave the `SIMULATION_MODE` setting as the place holder value, {}, the value that will be injected is `{"SIMULATION_MODE"="DEFAULT"}`.

To add a new symbol set in text, create a new file using an existing set as a template. For example: `SymbolSet-My3.evt`. Be sure to set its ID to correspond to the name.

Scenarios access new symbolsets by using the ""Symbol Attributes" SmartBlock distributed with this pack. This Smart Block will return all these values when given a SymbolSet identifier. See an example scenario (such as Iceberg) within the pack for usage. The simulation mode is also set via the symbolset. This is done by adding the simulation mode key to the extra parameters. For example:

```
com.apama.ata.SymbolSet("My3","","","", {},"","", "","",{},
   ["MSFT","AAPL","APMA"],["MSFT","AAPL","APMA"],
 { "SIMULATION_MODE":"BASIC"})
```

The ATA has four simulation modes, distinguished by their keyword:

- **BASIC** — A realistic simulator that includes an orderbook and a matching engine. It includes a built-in orderflow generator that simulates an orderflow to fill the book.
- **MARKETDATADRIVEN** — A simulator that includes the ability to feed market data, infer order flow from it, and then run that orderflow through the matching engine and orderbook.
- **OLDSTYLE** or **DEFAULT** — The BTexchange simulator from early releases of this solution. It is the default setting.
- **NONE** — No simulator is to be used.

4 Troubleshooting

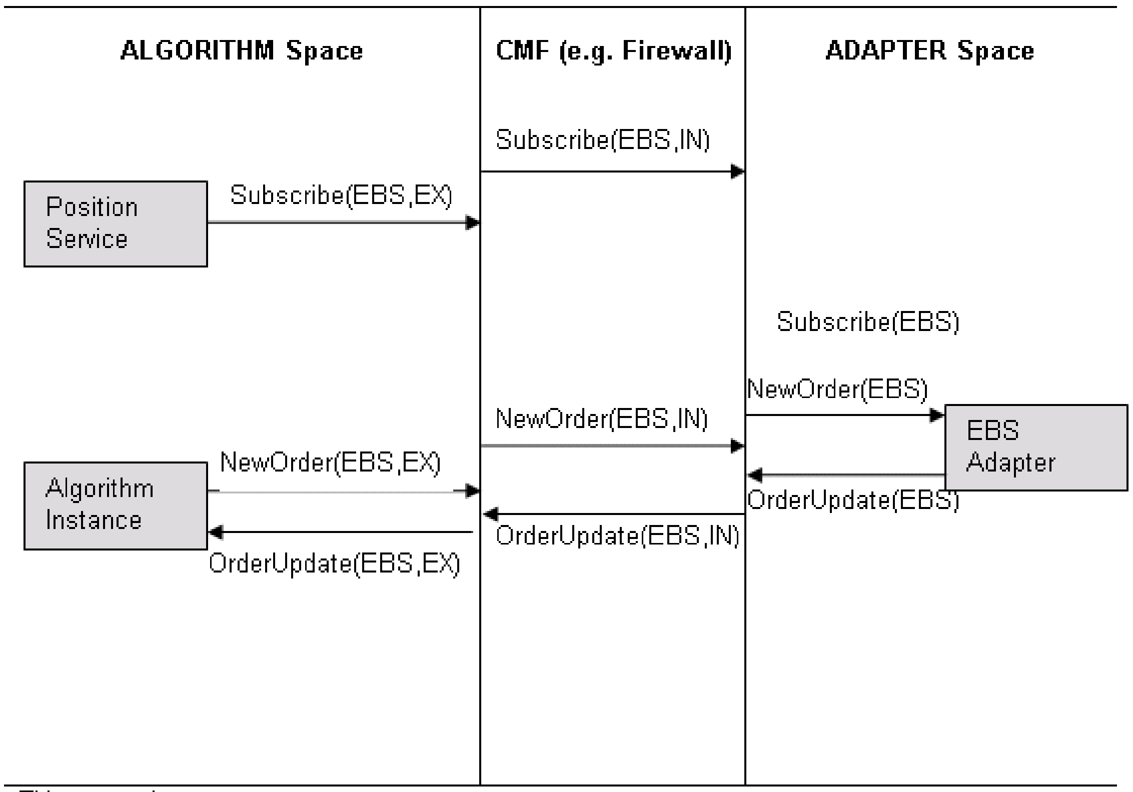
The following items are issues you might encounter:

There are symbols in the OMS screen, but all the prices are zero. The symbols appear in the OMS screen from a “scenario” level and request data on the Service and Market ID specified in the symbol set event files. The symbols appear in the Dashboard as a result of a request for data. If the end system (simulator monitor, IAF and service monitors) are not loaded or functioning correctly then all values will be zero as no price updates are being received. If the back end system has been verified as functioning correctly ensure that the service and market values defined in the symbols set EVT files are specified correctly.

Orders have been submitted but they do not reach the market. Orders appear in the blotter as the result of the OMS screen or an other scenario issuing a request to submit an order. The blotter table intercepts these requests and presents them on-screen. Thus, a request to submit an order is presented; but if that order does not reach the market then no updates will be received and thus no updates will be visible. This is possibly due to an incorrectly configured IAF/Service Monitor or incorrect values entered in the Symbol Set EVT definition files.

Symbol Set attributes are not returning values. Move the Symbol Attributes SmartBlock to the bottom of the column in the Scenario Manager, click **Deploy**, and then restart the Algorithmic Trading server.

Marketdata and OMS data needs to go through a firewall or other CMF boundaries.
All marketdata events (such as Depth, SubscribeDepth, Tick) that have the service-Id : `__SimpleMarketDataFirewallGatewayExternal` and all OMS events (such as NewOrder, AmendOrder) that have the service-id: `__ObjectionBasedFirewallControllerExternal` go through the firewall and other CMF gateway boundaries. Once the CMF services have dealt with these events, they replace the event’s service Id with the value given in the extraParameter: `Firewall.TargetService` and route the modified event.



EX: external
IN: internal
EBS: example service
e.g. NewOrder(EBS,EX) means serviceId = "__ObjectionBasedFirewallControllerExternal and extraParam Firewall.TargetService = EBS

Note: Replay logs. The `project.properties` file at the root of `apama-work\ATA_{version}` is set to `input` to minimize the logs that can accrue while exploring sample data . When directed by support to produce a replay log, modify the properties file as follows:

```
# ========================================
# Replay logging mode. Log is in the ${apama.work}/logs folder
#
REPLAY_LOG_MODE=replay
```

To learn more about replay logs, see *Deploying and Managing Apama Applications*.