TimeBase

Introduction to TimeBase Community Edition

Presenters:

Ilya Gorelik –
VP, Real-Time Computing Lab, EPAM
Ilya_Gorelik@epam.com

Aliaksei Vavilau –
Chief Architect, TimeBase,
EPAM
Aliaksei_Vavilau@epam.com

https://timebase.info/

Agenda

Presentation:

- Overview & History
- Architecture
- Technical details

Demonstrations:

- TimeBase Web Admin
- Examples Use-Cases
    - Portfolio Risk Monitor
    - Instant Payment Dashboard
- QuantHub Custom Research Environment
- Grafana and Jupyter (Source code round-trip)

Q&A

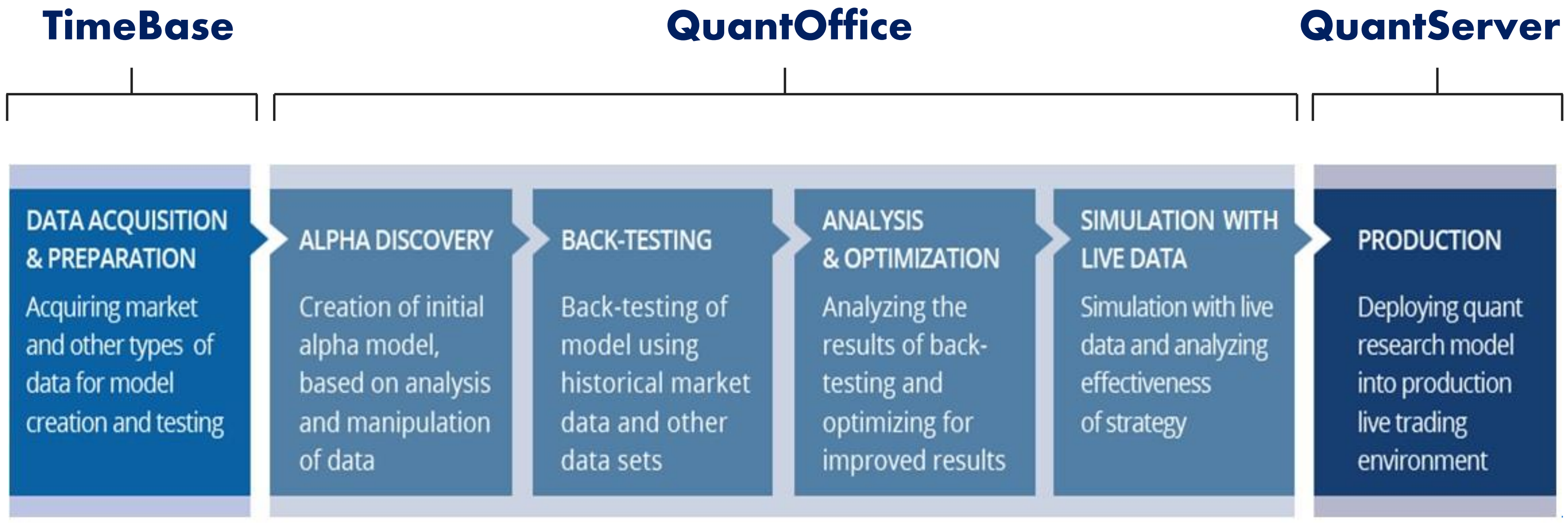
Overview

- **TimeBase** is a high-performance ***time series*** database and ***streaming system*** developed by EPAM Real Time Computing Lab (formerly Deltix, Inc.).
- Current TimeBase is a result of **15 years** of experience in financial domain. TimeBase runs standalone or in a cluster, processes millions of messages per second, stores terabytes of data, and can offer low microsecond latencies.

History

- 2006 – TimeBase 1.0 (historical data analytics, SQL)
- 2010 – TimeBase 2.0 (timeseries data base, live streaming)
- 2015 – TimeBase 5.0 (very large databases, HDFS)
- 2018 – IPC/UDP low latency mode (TimeBase topics)
- 2019 – Cloud support (REST/WS, Docker, Kubernetes)
- **2020 – TimeBase "Community Edition" (Open Source version); public website Timebase.info launched**

RTC Lab solution for advanced systematic trading


TimeBase
QuantOffice
QuantServer
DATA ACQUISITION & PREPARATION
Acquiring market and other types of data for model creation and testing
ALPHA DISCOVERY
Creation of initial alpha model, based on analysis and manipulation of data
BACK-TESTING
Back-testing of model using historical market data and other data sets
ANALYSIS & OPTIMIZATION
Analyzing the results of back-testing and optimizing for improved results
SIMULATION WITH LIVE DATA
Simulation with live data and analyzing effectiveness of strategy
PRODUCTION
Deploying quant research model into production live trading environment

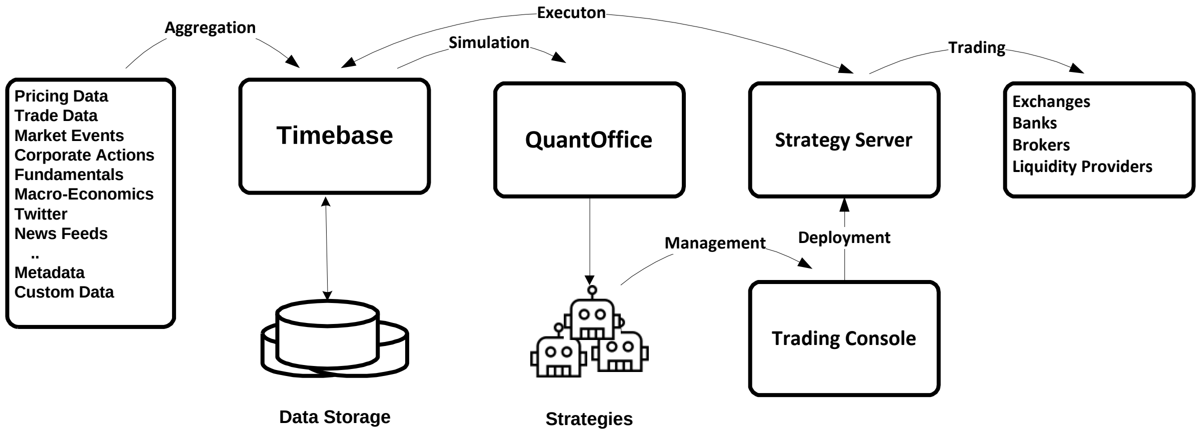
Source: Aite Group

RTC Lab at a glance: < aggregate | analyze | act >

	AGGREGATE	ANALYZE	ACT
EPAM TECHNOLOGY	**TimeBase** Enterprise-grade high-performance messaging middleware, time-series data warehouse and data steaming platform	**QuantOffice** Set of development tools and libraries for the creation, historical simulation and optimization of quantitative models	**QuantServer** High-performance ultra-low latency platform for the real-time live deployment, management and hosting of quantitative models
CORE TASKS AND ACTIVITIES	• Collect and aggregate data from many sources • Normalize and validate data • Maintain data archive for subsequent analysis • Stream data to research and real-time models	• Convert quantitative ideas to executable models • Test it on historical data, analyze results, optimize and assess predictive power • Simulate these models with real-time data	• Deploy custom and standard quant models in production to generate various deliverables: • Alerts, surveillance and risk signals • Reports • Trades and orders

TimeBase Use Cases

- Time-Series Data analysis
- Algorithm Back-testing
- Live Data streaming / Algorithmic Trading
- Warm-up/Live hybrid
- Market Data Aggregation / Ticker plant
- Messaging Middleware / Streaming Middleware
- Real time event processing
- Message Bus
- Complex Event Processing
- Long distance streaming



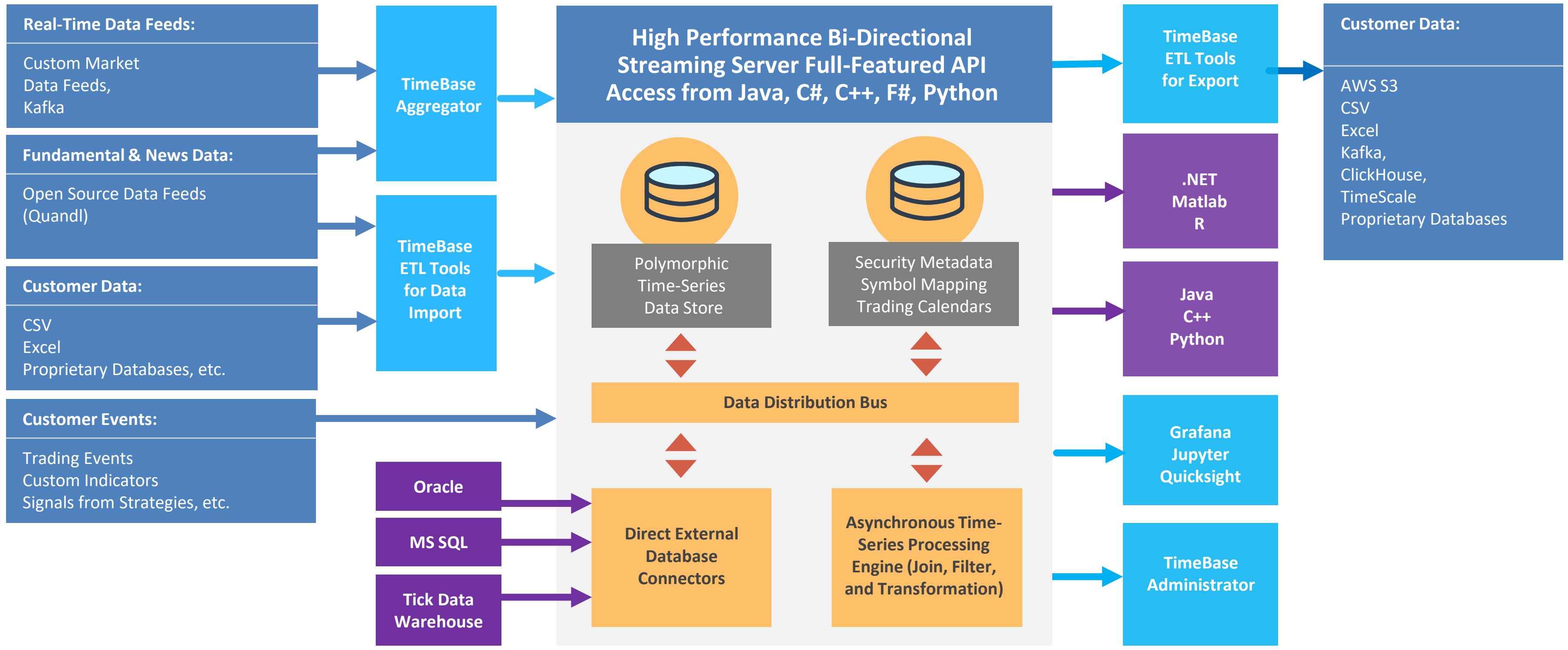
Key Features

- **Battle-tested.** 15 years in production with various implementations in the domain of financial markets
- **Cost-efficiency.** Impressive overall TCO and keeps the cost under control as system scales. The minimal fully functional installation is available on a single desktop/cloud instance, linearly scales to a large cluster. Doesn’t require special hardware, learning new language or programming techniques. Automated maintenance and data retention. The initial installation/deployment takes minutes.
- **Superior performance.** Million events per core per second and stable low microsecond latencies on community/cloud hardware.
- **Complete ecosystem.** Production ready integrations with various databases/message brokers/visualization tools (SQL Server, ClickHouse, TimeScale, S3, Kafka, Grafana, PowerBI and others). Connectors for 100+ market data venues.
- **Designed for analytics.** Supports both real-time and historical analytics, seamless transition between historical and live data. Native integration with Jupyter Notebook.
- **Multi-language API.** TimeBase provides API libraries natively implemented on most popular backend languages: Java/C++/C#/Python. Web APIs (REST/WS). Go API is under development.
- **Intelligent Data Modeling.** TimeBase is schema-oriented where the principle of introspection allows using classes in selected languages as schemas. TimeBase schemas allow to represent complex data models (like MBO or ITCH) utilizing features like polymorphic classes, arrays and nested objects. Essential data types are supported including nano-second timestamps, decimal floating-point numbers. Built-in performance-efficient serialization framework.

Key Technical Features

- Rich data schema formats, polymorphism, automatic codec generation.
- Complex Event Processing
- API: TimeBase provides the clients API for Java, C++, C#, and Python programming languages. REST / WebSocket consumer API.
- Queries: TimeBase offers SQL-like query language to retrieve data from single stream
- Support for HDFS / Azure Data Lake / SQLDB as alternative storage layer
- Connectors for 100+ market data venues, Normalized format to represent BBO / MBL / MBO market data.
- Data compression (GZIP) and data encryption (SSL)
- Support for UDP and IPC transport for local clients
- Monitoring solutions for Zabbix, JMX, SNMP
- Replication, Backup, Data Repair
- Deployed using installer, Ansible, or Alpine-based docker image
- Commodity hardware, on-premises or cloud, Linux / Windows

Architecture Diagram



Real-Time Data Feeds:
Custom Market Data Feeds, Kafka
Fundamental & News Data:
Open Source Data Feeds (Quandl)
Customer Data:
CSV
Excel
Proprietary Databases, etc.
Customer Events:
Trading Events
Custom Indicators
Signals from Strategies, etc.
TimeBase Aggregator
TimeBase ETL Tools for Data Import
Oracle
MS SQL
Tick Data Warehouse
High Performance Bi-Directional Streaming Server Full-Featured API Access from Java, C#, C++, F#, Python
Polymorphic Time-Series Data Store
Security Metadata Symbol Mapping Trading Calendars
Data Distribution Bus
Direct External Database Connectors
Asynchronous Time-Series Processing Engine (Join, Filter, and Transformation)
TimeBase ETL Tools for Export
.NET
Matlab
R
Java
C++
Python
Grafana
Jupyter
Quicksight
TimeBase Administrator
Customer Data:
AWS S3
CSV
Excel
Kafka,
ClickHouse,
TimeScale
Proprietary Databases

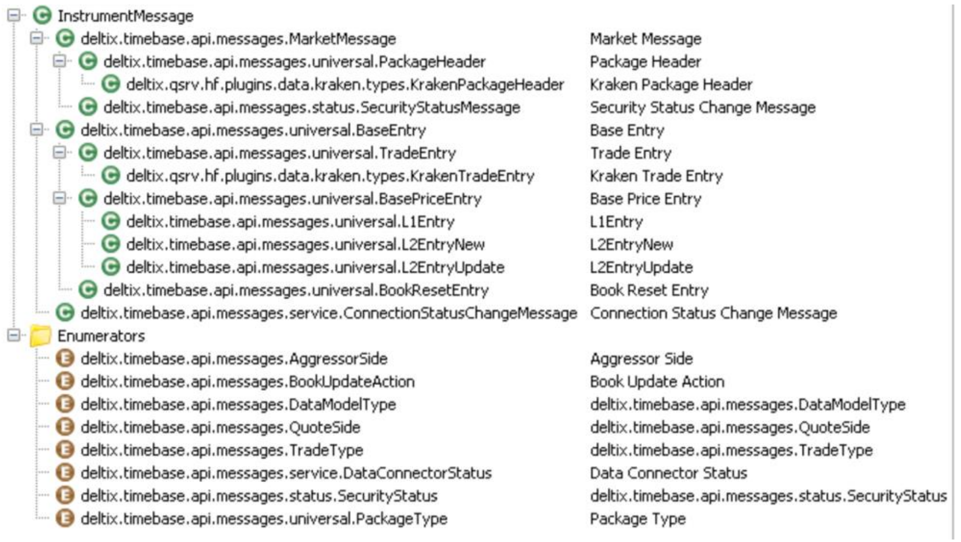
What Makes TimeBase unique

KEY DIFFERENTIATORS

- TimeBase offers real-time and historical data using **single** high-performance **streaming API**.
  Under the hood system may be tuned to stream data with sub-microsecond latencies or read/write millions of messages per second on each data producer and consumer. When streaming live data TimeBase can feed real-time consumers from memory buffers rather than disk (this feature provides significant latency reduction).
- While most of competitors offer key/value sets or BLOB messages, TimeBase natively supports complex **message structures that reflect Business Domain**. For example, TimeBase can store business layer messages that reflect Order Book snapshots or incremental updates, as well as Trade Order requests.

TimeBase Concepts

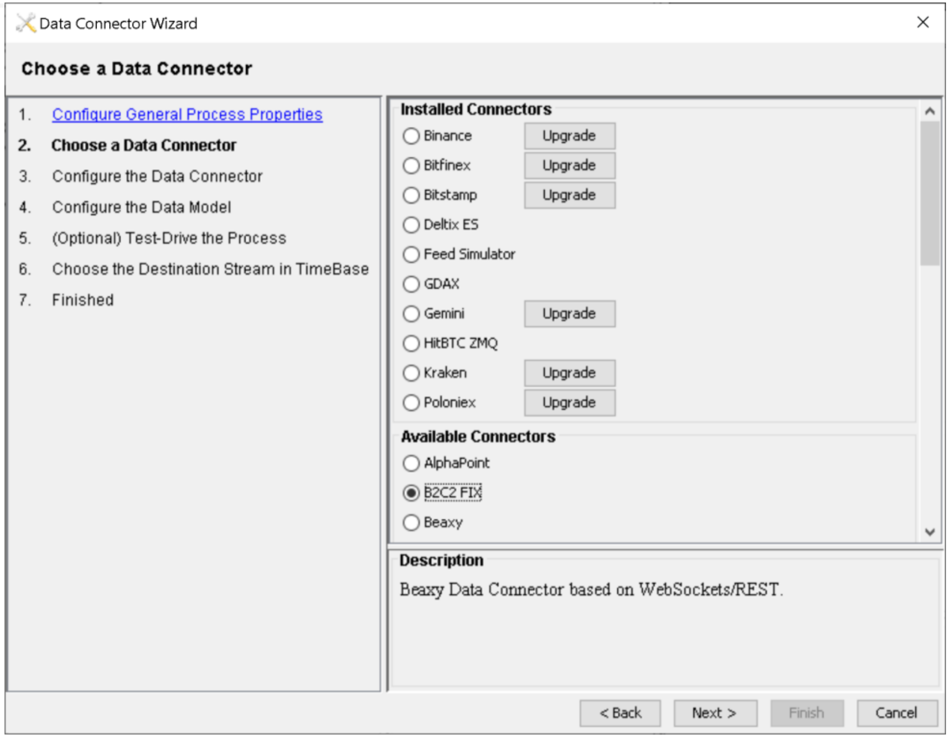
- TimeBase database has many **Streams**
- Each stream contains many **Messages** of one or several types
- Type of each message is defined by Schema. Each message has
    - Timestamp
    - Symbol
    - Some number of custom fields (simple or composite), according to schema
- Producers use “**Loaders**” (load data into TimeBase)
- Consumers use “**Cursors**” (iterate over data read from TimeBase)
- Stream types:
    - Durable (persistent)
    - Transient (in-memory only)
    - *Topics* (Bypass server)
- TimeBase has **QQL** Query Language to query data or describe/modify stream data schema.



```
InstrumentMessage
  deltix.timebase.api.messages.MarketMessage                        Market Message
    deltix.timebase.api.messages.universal.PackageHeader            Package Header
      deltix.qsrv.hf.plugins.data.kraken.types.KrakenPackageHeader  Kraken Package Header
    deltix.timebase.api.messages.status.SecurityStatusMessage       Security Status Change Message
  deltix.timebase.api.messages.universal.BaseEntry                  Base Entry
    deltix.timebase.api.messages.universal.TradeEntry               Trade Entry
      deltix.qsrv.hf.plugins.data.kraken.types.KrakenTradeEntry     Kraken Trade Entry
    deltix.timebase.api.messages.universal.BasePriceEntry           Base Price Entry
      deltix.timebase.api.messages.universal.L1Entry                L1Entry
      deltix.timebase.api.messages.universal.L2EntryNew             L2EntryNew
      deltix.timebase.api.messages.universal.L2EntryUpdate          L2EntryUpdate
    deltix.timebase.api.messages.universal.BookResetEntry           Book Reset Entry
  deltix.timebase.api.messages.service.ConnectionStatusChangeMessage  Connection Status Change Message
Enumerators
  deltix.timebase.api.messages.AggressorSide                        Aggressor Side
  deltix.timebase.api.messages.BookUpdateAction                     Book Update Action
  deltix.timebase.api.messages.DataModelType                        deltix.timebase.api.messages.DataModelType
  deltix.timebase.api.messages.QuoteSide                            deltix.timebase.api.messages.QuoteSide
  deltix.timebase.api.messages.TradeType                            deltix.timebase.api.messages.TradeType
  deltix.timebase.api.messages.service.DataConnectorStatus          Data Connector Status
  deltix.timebase.api.messages.status.SecurityStatus                deltix.timebase.api.messages.status.SecurityStatus
  deltix.timebase.api.messages.universal.PackageType                Package Type
```

Data Ingestion

- Market Data Connector API (100+ connectors developed)
- Aggregator = Manager for connectors
- Simple API clients (Java, C#, Python)
- Normalized Market Data format (L1/L2/L3)



Data Connector Wizard

Choose a Data Connector

1. Configure General Process Properties
2. **Choose a Data Connector**
3. Configure the Data Connector
4. Configure the Data Model
5. (Optional) Test-Drive the Process
6. Choose the Destination Stream in TimeBase
7. Finished

Installed Connectors

○ Binance	Upgrade
○ Bitfinex	Upgrade
○ Bitstamp	Upgrade
○ Deltix ES	
○ Feed Simulator	
○ GDAX	
○ Gemini	Upgrade
○ HitBTC ZMQ	
○ Kraken	Upgrade
○ Poloniex	Upgrade

Available Connectors

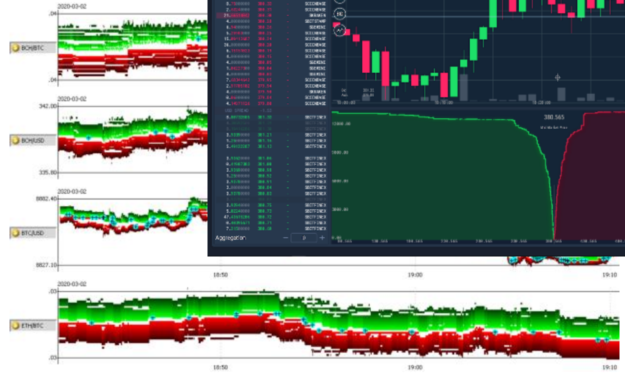
○ AlphaPoint
[✓ B2C2 FIX]
○ Beaxy

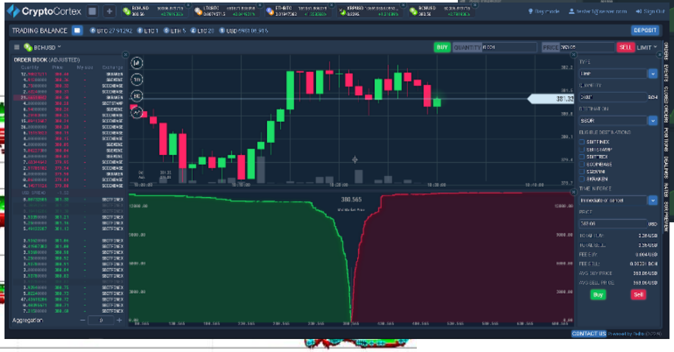
Description

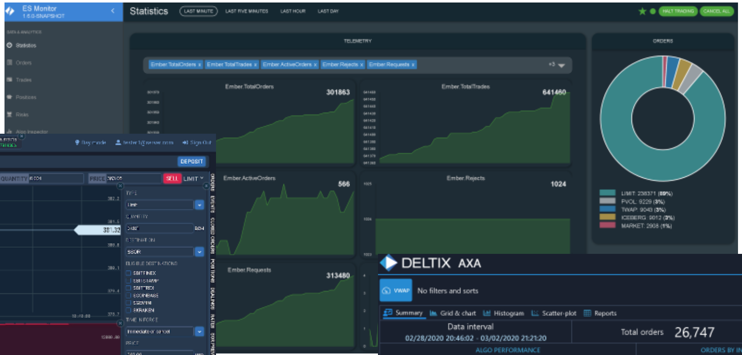
Beaxy Data Connector based on WebSockets/REST.

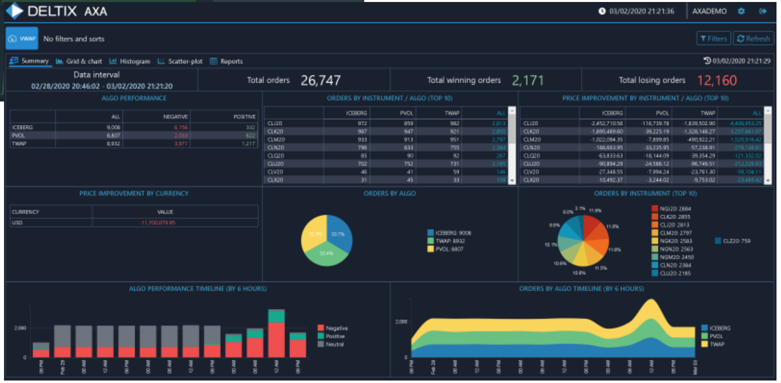
Data Processing

- Historical Data: Algorithm Back-testing, Quantitative Research: QuantOffice
- Live Market Data: Aggregation / Trading
- Custom API clients (Java/C#/Python/REST/WS)









DELTIX AXA
03/02/2020 21:21:36
AXADEMO
VWAP
No filters and sorts
Filters
Refresh
Summary
Grid & chart
Histogram
Scatter-plot
Reports
Data interval
02/28/2020 20:46:02 - 03/02/2020 21:21:20
Total orders 26,747
Total winning orders 2,171
Total losing orders 12,160
ALGO PERFORMANCE
ORDERS BY INSTRUMENT / ALGO (TOP 10)
PRICE IMPROVEMENT BY INSTRUMENT / ALGO (TOP 10)
PRICE IMPROVEMENT BY CURRENCY
ORDERS BY ALGO
ORDERS BY INSTRUMENT (TOP 10)
ALGO PERFORMANCE TIMELINE (BY 6 HOURS)
ORDERS BY ALGO TIMELINE (BY 6 HOURS)
ICEBERG
PVOL
TWAP
Negative
Positive
Neutral

Performance

- Throughput:
  - TimeBase Streams (TCP)
    - 1 Producer x 1 Consumer = 1.8 M messages/sec
    - 1 Producer x 4 Consumers = 5.5 M messages/sec
  - TimeBase Topics (100 bytes payload)*
    - 1 Producer x 1 Consumer = 9 M messages/sec
    - 1 Producer x 4 Consumers = 24 M messages/sec (6M per consumer)
- Latency (one way, microsecs.):

Streams

```
# Mean = 49.2 (StdDev = 120)
Percentiles:
  50% = 46
  90% = 56
  99% = 62
  99.9% = 66
  99.99% = 7065
  99.999% = 8781
  Max    =        9822
```

Topics

```
# Mean = 0.372 (StdDev = 0.186)
Percentiles:
  50% = 0.356
  90% = 0.373
  99% = 1.404
  99.9% = 1.484
  99.99% = 1.643
  99.999% = 4.735
  99.9999% = 34.239
  Max = 130.943
```

*See TimeBase Topics presentation for more information

Hardware / Software Requirements

HARDWARE

- Can run on single laptop or server farm. Commodity hardware
- Back-testing – more RAM is good for cache (minimum 0.5G)
- HDD/SSD/NVME – depends on use case (from 100Mb to 100Tb)
- CPU – depends on use case (minimum 2)

SOFTWARE

- Windows 10 / Windows Server 2019
- Linux (CentOS, RHEL, Amazon Linux, Ubuntu)

CLOUD

- Ansible
- Docker / Docker-compose
- Kubernetes