exactpro
EXITUS ACTA PROBAT

LTB 2026
Load Testing & Benchmarking

Presentation Track

Beyond message injection: advanced load testing capabilities for trading systems with th2-shark:

A framework for benchmarking, latency analysis, and functional validation under load

Alyona Bulda, SVP, Emerging Technologies, Exactpro

4 May 2026

BUILD SOFTWARE TO TEST SOFTWARE
exactpro.com

Agenda

- Typical trading systems
- Distributed and non-deterministic
- th2-shark – actor testing
- Real-time monitoring
- Latency analysis
- Reconciliation
- Exploratory testing

Meet the author

Alyona has been leading diverse teams of software testing and development engineers, business analysts, and DevOps specialists across time zones and agile streams for over 15 years – facilitating software quality excellence for leading FMIs and financial technology providers.

Client systems that Alyona has worked with have included a leading European multilateral **trading facility, one of the world's largest interdealer brokers in OTC financial and commodity related products,** a premier technology solutions provider, a leading global **rates and multi-asset clearing house and a multi-national central counterparty**. She has also managed software testing of LSEG's **FPGA-accelerated Market Data Dissemination** platform.

Alyona's area of expertise spans project management and delivery of large-scale technology transformations related to both traditional and decentralised financial instruments **(TradFi and DeFi domains, respectively)**, the latter including Central Bank Digital Assets issuance and distribution. A deep understanding of both established systems and emerging technologies informs Alyona's consultancy and project management roles, enriching them with a unique perspective.

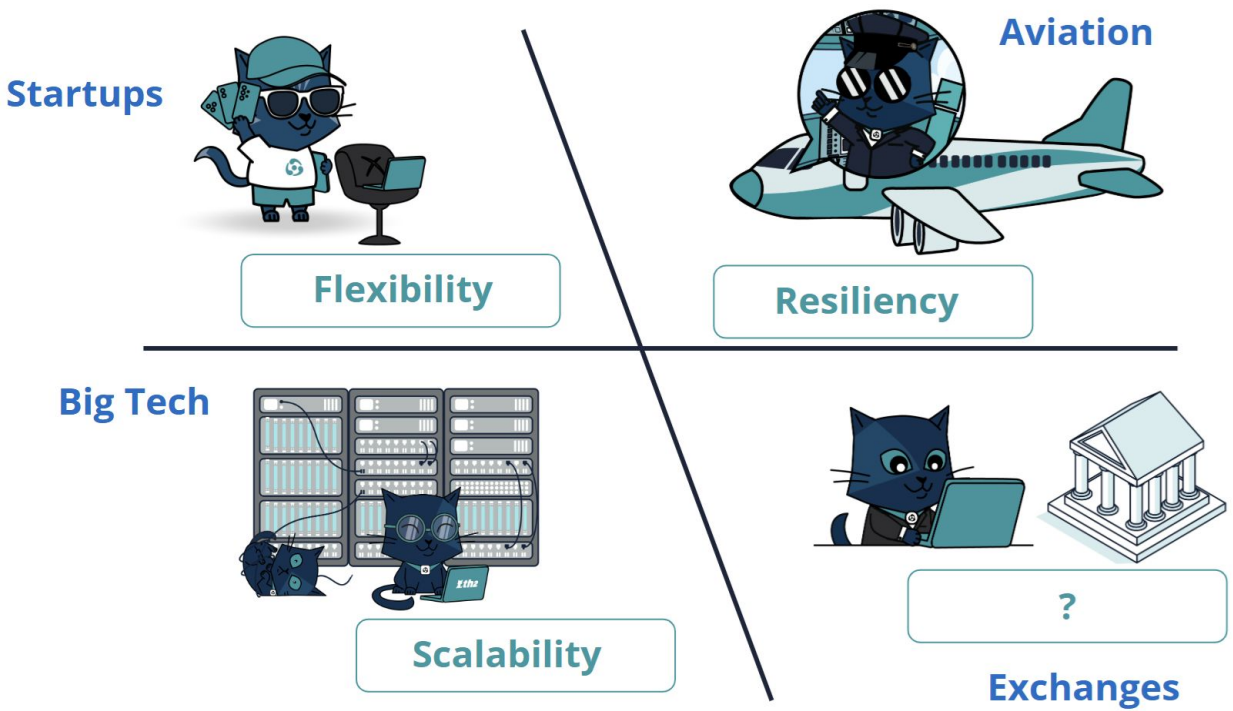
With experience covering the full enterprise **software delivery lifecycle**, Alyona has an advanced skill set for analysing complex trading, market data and post-trade solutions, with a focus on test strategy development, integration testing, test execution, automation approaches and regulatory compliance.

Alyona has been combining work on software testing projects for Exactpro clients and R&D work, having authored and co-authored several research papers, including some published by the IEEE Computer Society.

Alyona Bulda

Emerging Technologies, Exactpro

Trading Systems / Stock Exchanges and Other Systems

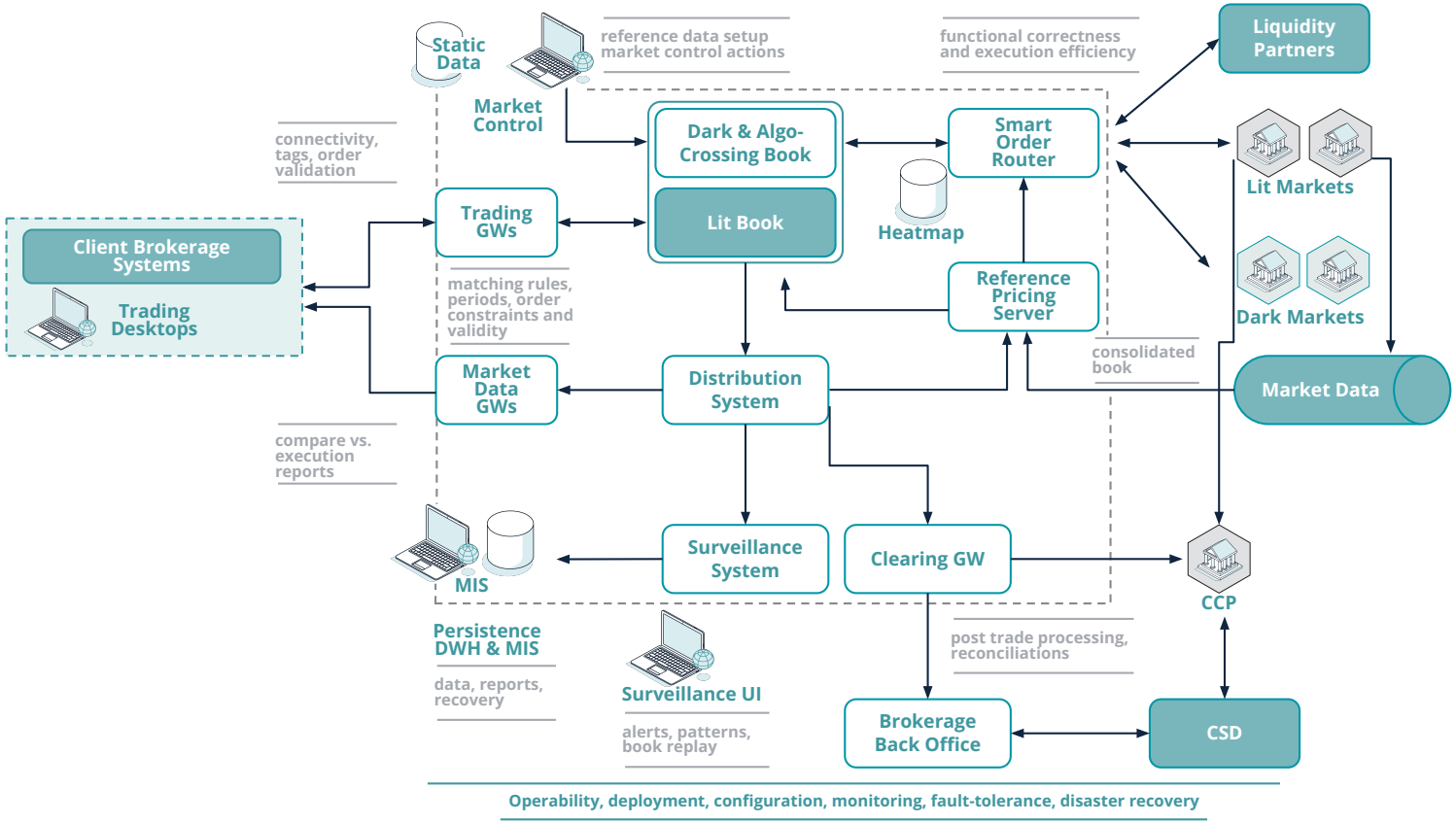


Challenges and complexities

- Principle- and outcome-based regulatory frameworks
- Expectation of robust governance and operational resilience
- Technological and epistemological complexity
- Massive transformations
- Adopting the new, aligning with existing and ecosystems
- High visibility and impact on national economy and society
- Budget and time constraints
- Centralisation and no canary deployments

Typical Trading Systems

exactpro
EXITUS ACTA PROBAT

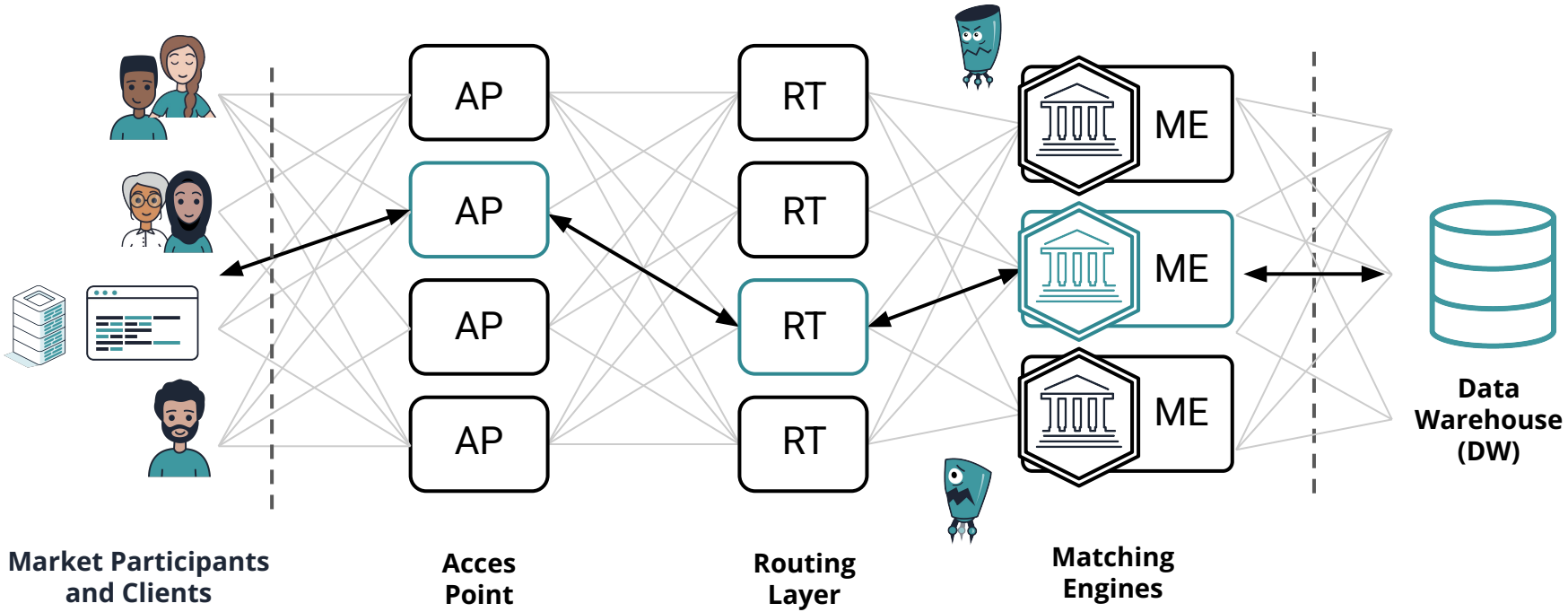


Non-functional Requirements for Trading Systems

exactpro
EXITUS ACTA PROBAT

Non-functional requirement	Description
Low-latency order entry and trading processing	Providing fast processing of the placed orders, quotes and their matching algorithms.
Real-time order entry responses, event updates, market data dissemination, request processing	Real-time order entry responses, market data dissemination, including the enriched and calculated (like Trade Volume, High/Low Trade Prices, etc.) market data updates and fast processing of requests received from clients and quotes data.
Mass events through daily/weekly life cycles	Maintaining and processing dynamic mass events in the main trading components (like matching engines) happening during the trading schedule with the load applied against the system.
Continuous working efficiency of the system	Ensuring high uptime and working system availability through failure and recovery mechanisms
Operability of the trading system	Providing effective and efficient system management mechanisms, allowing proactive manual intervention where/if needed.
System monitoring (resource efficiency) and logging	The availability of applications to monitor the system and operate its components with alerts for any signs of failure or current issues, including the systems' non-functional characteristics such as performance, latency and throughput.
Providing throughput	Handling high volumes of data w/o performance degradation and applying the Fair Access Rule, especially during peak market activity.
Providing fault tolerance	Redundancy, failover and load balancing mechanisms.
Disaster recovery	Ensuring robust disaster recovery plans to handle data center failure events with minimal data loss. Recovery time evaluation.
Maintainability	During very high system uptime, system updates or patches must be within the limits of minimal downtime or disruption.

Distributed and Non-Deterministic

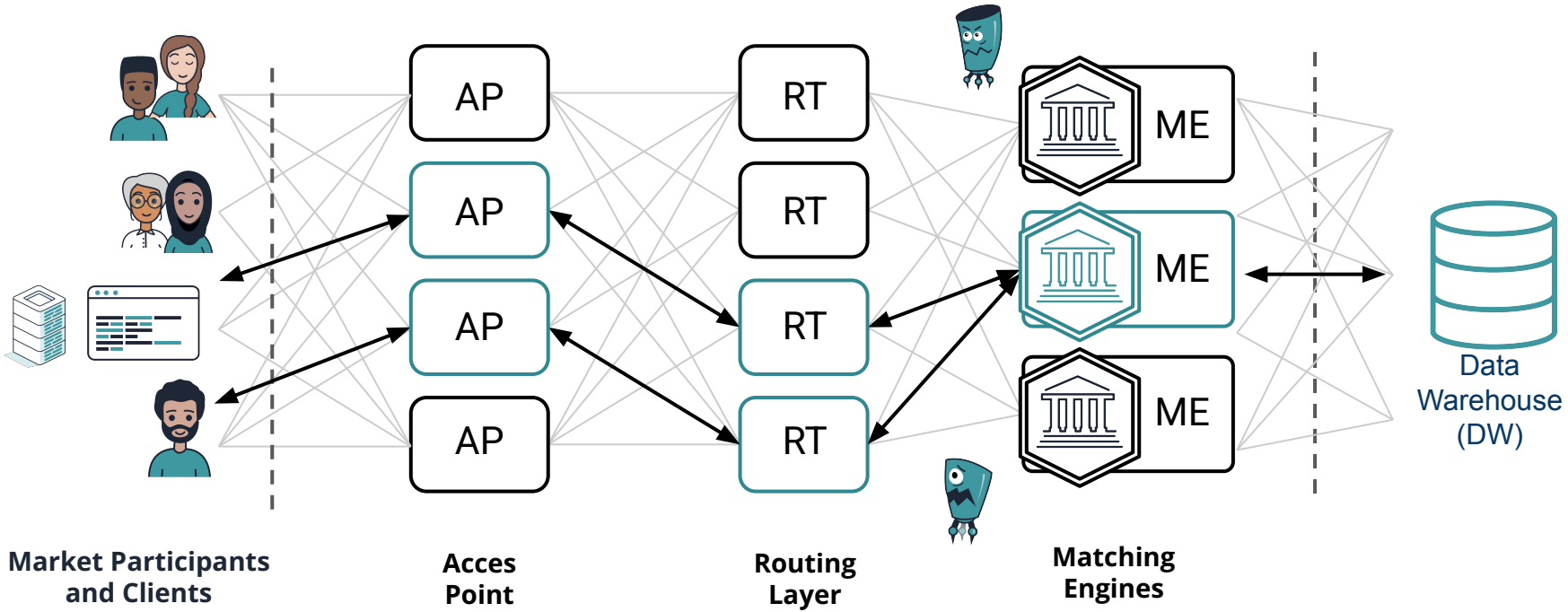


Infrastructure- and Non-functional Requirements-related Challenges

Due to the dynamic environment of electronic trading and the high structural complexity of its interconnected infrastructure and process flows, a high-performance trading system testing strategy should effectively account for the following **challenges**:

- High data volumes that are prone to fluctuations
- Ultra-low latency real-time performance
- Possibility of data loss or corruption
- Possibility of interrupted delivery of data
- Time synchronization across the infrastructure
- Integrations with surrounding systems

Distributed and Non-Deterministic



Sources of Non-determinism

In a complex distributed system, it is crucial to understand the possibility of hidden interdependencies that will only reveal themselves at the intersection of functional and non-functional testing. Some potential situations of non-determinism occurring under specific non-functional conditions can be found in the below scenarios:

- Any complex system must sustain a certain level of **concurrency** especially when a high-frequency load is applied.
- **Production-like randomisation** combined with load conditions.
- Integer **overflow** in certain scenarios, requiring calculation and ID incrementation.
- **Dynamic mass events** in the main trading components happening during the Daily Life Cycle (DLC) with the load applied against the system.

Latency and Throughput Requirements

exactpro
EXITUS ACTA PROBAT

Daily capacity - **billions of transactions**

Peak rates - **from hundreds of thousands to millions of transactions per second**

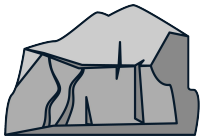
Average round-trip latency - **dozens of microseconds**

> 3,000 trx

2.5 cm

<1 mm

History of Tools



C++
LoadInjector

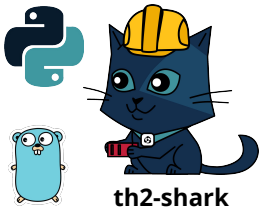
16 years ago: Just works

+ relatively fast in simple cases
- business logic is hard-coded by developers
- very complicated process of load scenario preparation
- requires many dependencies, special build environment and much time for building
- cannot be used for anything but load testing
+- relatively low footprint

Sailjector

8 years ago: Work smarter

+- fast enough to do projects without high load requirements
+ business logic can be written by testers
+ load scenarios preparation is easier and faster
+- quick updates, can run from any machine with portable Python
+ can be used for all types of testing, including functional and reconciliation testing
- relatively high footprint



th2-shark

Now: Work faster

+ fast to do all our projects
+ business logic can be written by testers
+ load scenarios preparation is even easier
+ builds instantly, can run from any machine without any dependencies
+ can be used for all types of testing, including functional and reconciliation testing
+ low footprint

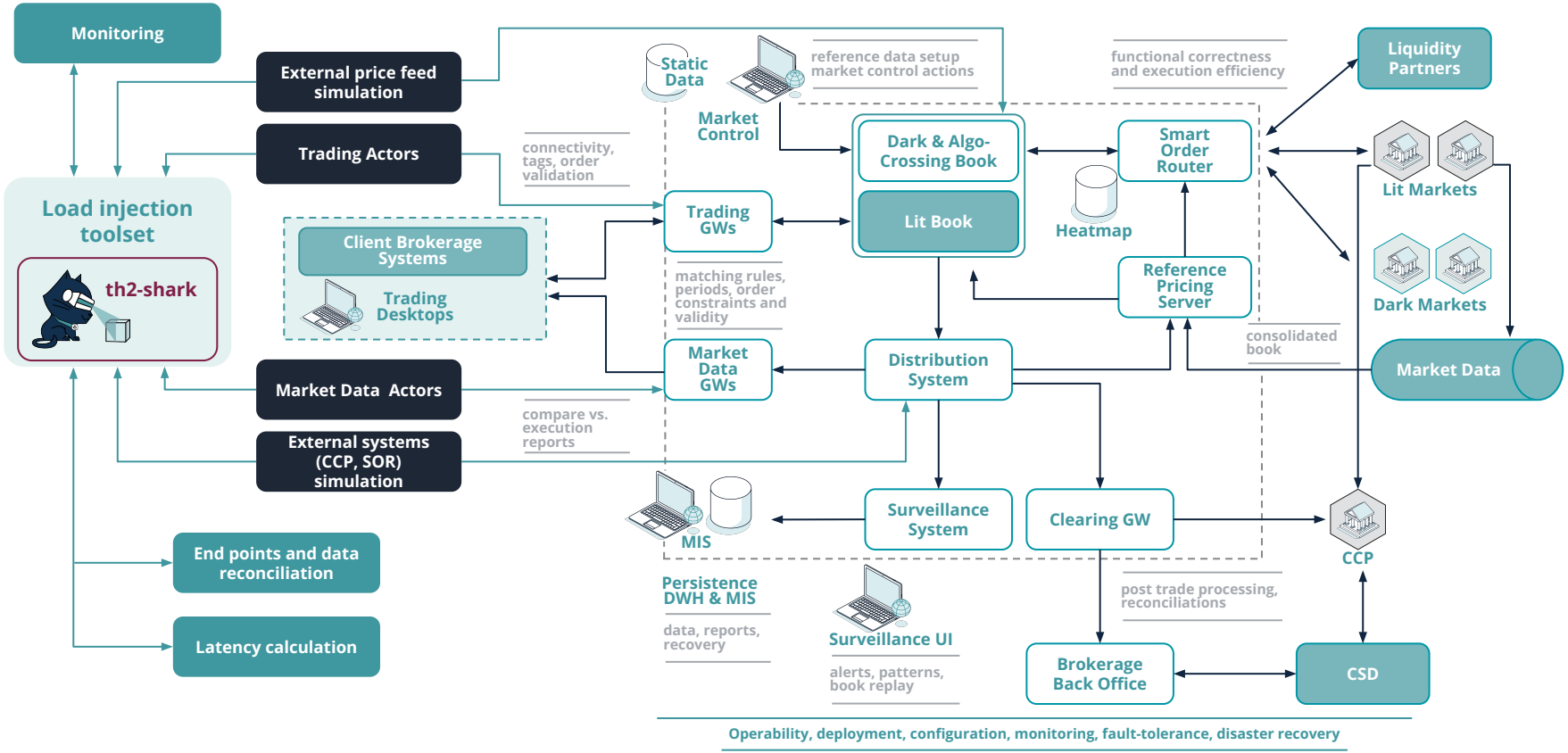
th2-shark in the Industry Context

High-performance trading systems operate in environments where microsecond-level latency differences determine competitive outcomes and where instability under load is as damaging as functional defects. Validating such systems demands load testing tooling purpose-built for the unique demands of electronic trading infrastructure capable of sustaining millions of messages per second, simulating realistic market participant behavior, supporting a wide range of financial protocols, and producing actionable, analyst-ready performance reports.

th2-shark is a high-throughput load generation and benchmarking tool designed specifically for these requirements. Its core **design principle** is that test scenarios should be **expressive, portable,** and impose **minimal overhead** on the system under test.

It can be deployed against **matching engines, smart order routers, market data systems, clearing** and **settlement infrastructure,** and **risk management** platforms without protocol-specific tooling gaps.

Typical Trading Systems – Load Injection with th2-shark



th2-shark Capabilities

exactpro
EXITUS ACTA PROBAT

th2-shark is an **actor testing framework** for non-functional and functional testing with the following capabilities:

- a standalone statically built binary ready to go
- no 3rd-party dependencies
- minimal footprint
- high throughput
- easily extendable
- able to simulate production-like trading conditions (complex scenarios under high load)
- simultaneously capture performance telemetry (monitoring)

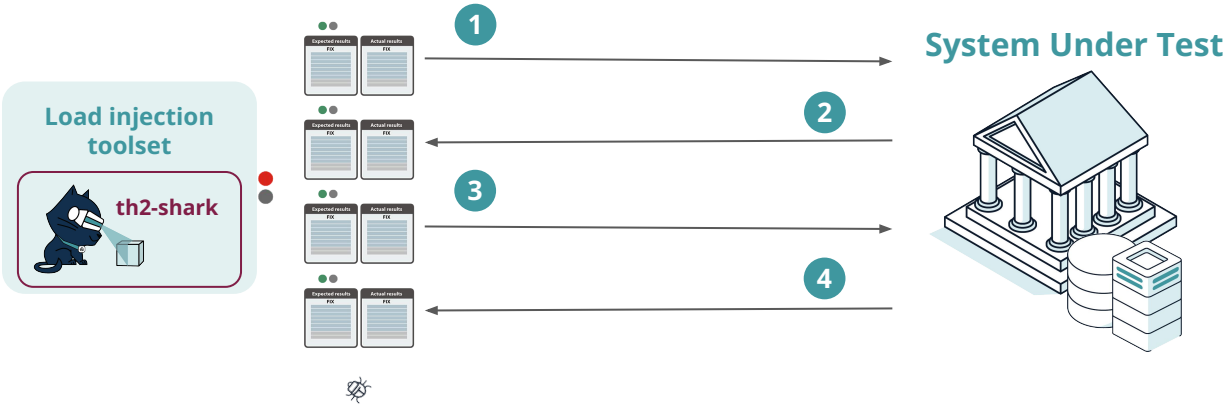
th2-shark provides comprehensive **coverage of the protocols** used in financial market infrastructure:

- TCP/IP and SOUP-based: OUCH, SOUP ITCH, GLIMPSE, LSE Native
- UDP and MoldUDP-based: Mold ITCH, REWIND, MIT ITCH, FAST
- FIX variants: FIX, FIX SBE (Simple Binary Encoding), FAST

th2-shark – Actor Testing

Actor testing is a hybrid (active and reactive) model where every entity acts in accordance with changing conditions.

Typical scenario: an actor sends a message (e.g. order), receives a response from the system under test (e.g. market data) and prepares the next message based on the new information (e.g. updates order price). The actors may act independently or cooperate with others.



th2-shark - Configuration

Load testing with monitoring:

```shell
./loader -c config/main.yaml -r "(100:60,200:120)*99" -R "localhost:18000" -M -B
```

Print traffic to JSON: after executing one of the above examples, the traffic directory may contain binary traffic files:

```shell
./loader -e PrintFix -V WASMMODULE="examples/exch/wasm/printer_plugin.wasm" -V
TRAFFIC="traffic/*.dat" -H
```

Alternatively, a PCAP file may be used:

```shell
./loader_pcap -e PrintFix -V WASMMODULE="examples/exch/wasm/printer_plugin.wasm"
-V TRAFFIC="traffic/*.pcap" -H
```

th2-shark – Configuration

exactpro
EXITUS ACTA PROBAT

JSON

```
ActorGroups:
  - Name: FIX_SERVER
    LoadPercent: 0
    Actors: configs/fix_server.yaml
    Range: 1

  - Name: FIX
    LoadPercent: 100
    Actors: configs/fix.yaml
    Range: 1
    Options:
      FixClient:
        TCPClient:
          AddressList:
            - "127.0.0.1:1234"
```

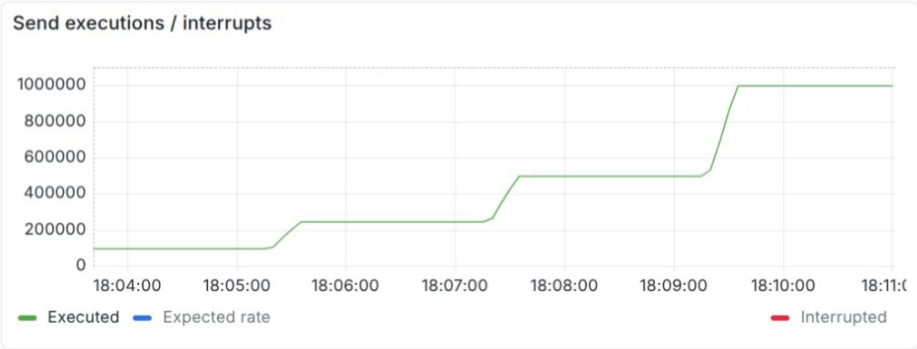
JSON

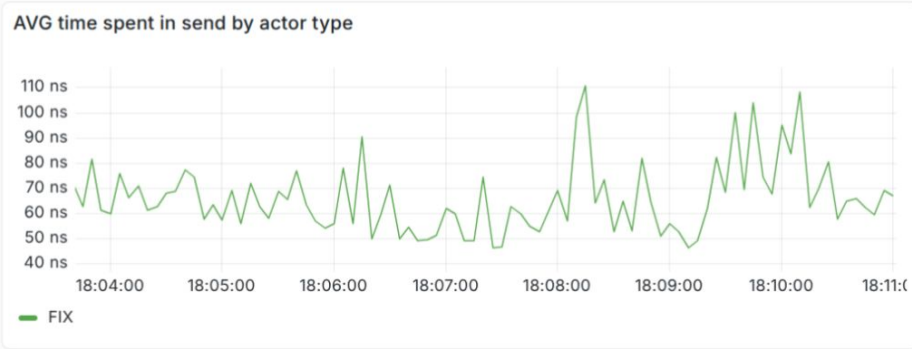
```
- Type: FIX
  Name: DEMO
  Options:
    WASM:
      Module: "wasm/client_plugin.wasm"
      Config:
        SenderCompID: "USER"
        TargetCompID: "SERVER"
        HeartBtInt: 10
    FixClient:
      TimeZone: UTC
      TimestampFormat: "20060102-15:04:05.000"
      LogonTimeout: 30
      Traffic:
        In:
          FileName: traffic/${date}/fix/${session}_${time}_${index}.in.dat
          WriterId: DEMO
        Out:
          FileName: traffic/${date}/fix/${session}_${time}_${index}.out.dat
          WriterId: DEMO
      TCPClient:
        ConnectAttempts: 5
        RetryCurrentAddress: true
        ReconnectAttempts: 5
        ReconnectInterval: 3000
        AddressList: ["127.0.0.1:9996", "127.0.0.1:9997"]
```

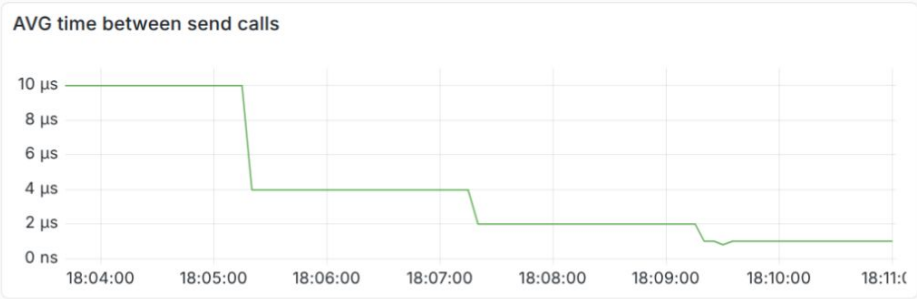
Real-Time Monitoring

A generic dashboard with internal metrics to validate outbound load rate and actors' performance. It's also possible to expose custom metrics, e.g. to monitor round-trip latency in real time.

Additional monitoring with the correlated system view (CPU / network / rate).







AVG time spent in send by actors

Name	Type	Time ↓	Interrupts	Executions
34	FIX	325 ns	0	2009902
40	FIX	194 ns	0	2009902
20	FIX	85.4 ns	0	2009902
1	FIX	82.4 ns	0	2009903

Latency Analysis – Traffic Data Post-processing

The tool stores inbound and outbound traffic in binary files that can be processed after the execution. Also, can work with network capture (PCAP).

A typical example is building latency reports:

Metric	Max, us	90th, us	Mean, us	Count	Percent
Cancel to ExecutionReport	35602	1754	65	218545	33.029
New to ExecutionReport	33363	1803	74	432273	65.33
Replace to CancelReject	3829	3829	3829	1	0
Replace to ExecutionReport	26641	1926	125	10861	1.641
TOTAL	35602	1789	71	661680	100

High Dynamic Range (HDR) histogram:

Metric: Cancel to ExecutionReport

From, us	To, us	Count	Percent
50	51	33	0.015
52	53	404	0.185
54	55	2608	1.193
56	57	8475	3.878
58	59	16685	7.635
60	61	24149	11.05
62	63	28480	13.032
64	67	52425	23.988
68	71	27923	12.777
72	75	9544	4.367
76	79	3187	1.458
80	83	1262	0.577
84	87	599	0.274

...

Reconciliation – Traffic Data Post-processing

exactpro
EXITUS ACTA PROBAT

The below table shows comparison of 2 sequential test runs using captured internal system traffic. The tool was replaying production traffic for the same date vs 2 different releases of the system under test.

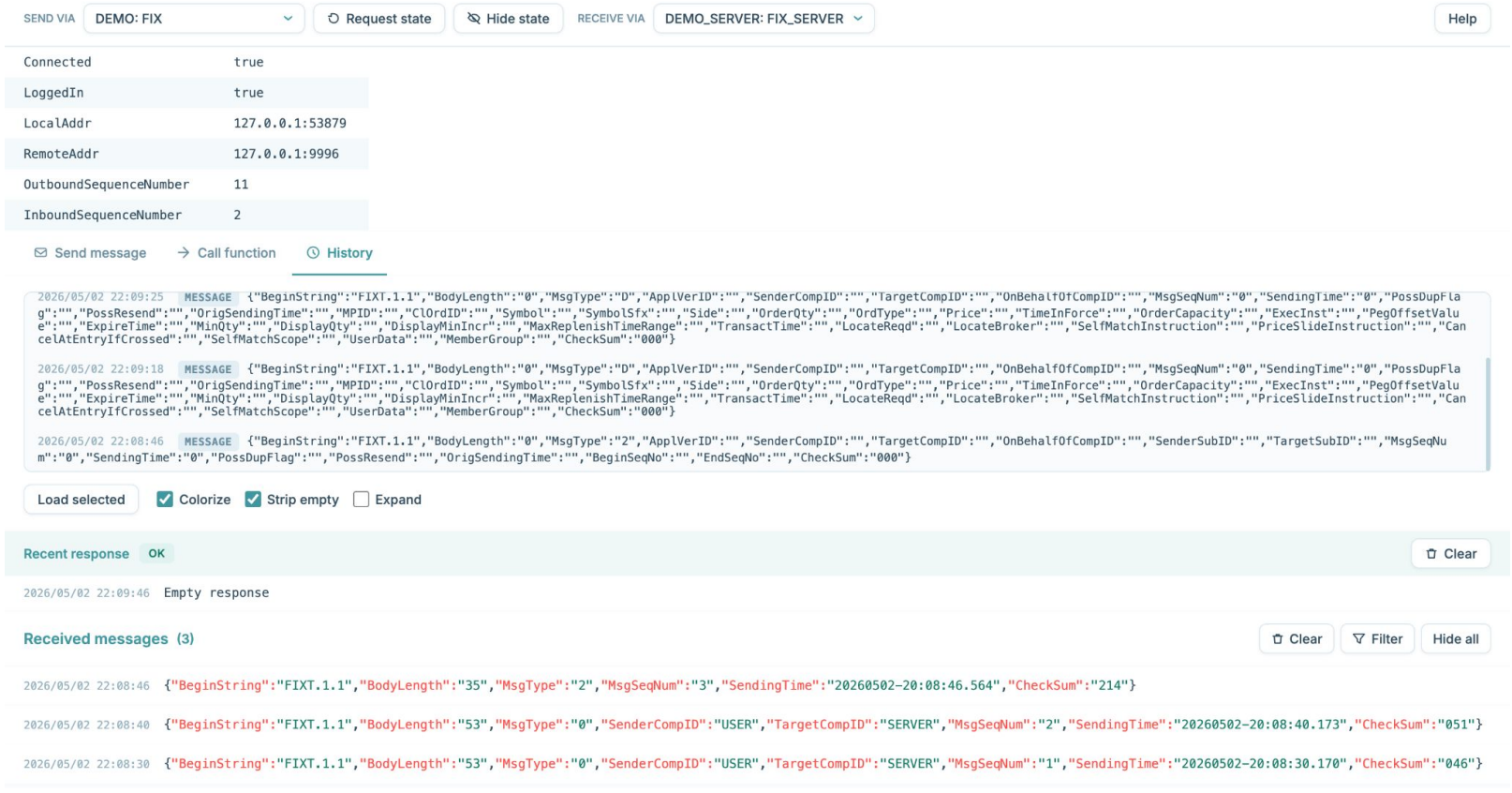
Symbol: XXX

Key	Source value	Replay value
MessageType	10	10
TimeCreated	1668529187829322952	1671018382005816359
TimeChanged	1668529198883087412	1671018392429746563
OrderBookId	70571	70571
TriggerOrderBookId	0	0
ParticipantId	75	75
UserId	4917920	4917920
OnBehalfOfSubmitterId	0	0
OrderId	7644027766576353830	7654566860412063465
PreviousOrderId	0	0
ClientOrderId	50	50
Side	1	1
Price	2500	2500
OrderQuantity	1470	1470
LeavesQuantity	953	953
DisplayQuantity	953	517
MinimumQuantity	0	0
TimeValidity	256	256
OrderType	1	1
ExchangeOrderType	2048	2048
OrderCategory	1	1
AccountId	10	10

...

Exploratory Testing

exactpro
EXITUS ACTA PROBAT



Thank You!

If you have got a follow-up question, please reach out to

alyona.bulda@exactpro.com