Athens Stock Exchange Trading System FIX Migration

OASIS Upgrade Testing & Coverage Analysis

Contents:

1. Introduction

This case study focuses on the Athens Exchange ('ATHEX', 'exchange') Automated Integrated Trading System ('OASIS') platform upgrade and the related software testing support provided by Exactpro. During the upgrade, the legacy **ATHEX Gateway** of the OASIS platform previously supported by a COMM Server was to undergo a migration to a new **FIX Server**.

The business advantages of modernising the infrastructure include attaining cost efficiencies and strengthening the ATHEX ecosystem and its capital markets offering. The technical benefits of the new FIX Server included achieving enhanced security and recovery, improving throughput, performance, and scalability, while decreasing the complexity of the configuration and client connectivity, reducing latency, and minimising the amount of points of failure and spikes under load.

> "In our continuous pursuit of delivering efficient services, ATHEX is currently undergoing a significant IT initiative focused on the modernization of our software and hardware, with a particular emphasis on our trading platform. The primary objectives of this initiative are to eliminate legacy components, adopt industry standards and enhance performance with a clear focus on HFT (High-Frequency-Trading). Technology plays a crucial role in driving business advancements and, ultimately, increasing revenues and ATHEX is fully committed to enabling progress in this regard," **comments Theodoros Zarros, CTO, Athens Exchange Group**.
>
> "The introduction of ATHEX FIX, our new component supporting all order routing, marks the first bold step towards the modernization of our trading platform, which is undeniably a mission-critical task. This project takes the form of a new implementation, that fully replaces legacy software without impacting APIs and other interfaces. Our top priority is to ensure the validation of existing functionality. When stakes are high and room for error is non-existent, utmost care must be exercised to mitigate the risks. Automated, end-to-end functional testing becomes a borderline necessity and an absolute key factor in achieving the required levels of quality assurance."

In testing large-scale mission-critical financial technology, test library automation is effectively achieved via a combination of active and passive testing methods. Thus, the case study serves a dual purpose: 1) it provides a reference use case for supporting trading system migrations to FIX-enabled technology, and 2) highlights the role of passive testing approaches in performing and automating regression testing and improving test coverage.

2. Trading System Migration Scope

During the trading system migration, some of the functionality of the legacy ATHEX Gateway (ATHEX GW) of the OASIS platform was to be decommissioned. The new FIX Server was to take on new functionality.

From the testing perspective, the upgrade involved two main stages:
1) Recreating the ATHEX test environment and performing regression testing against the **legacy ATHEX GW** system to adjust the test library and bring it up to date. This allowed the Exactpro team to upgrade the Test Framework to the latest versions of the ATHEX GW components released since the earlier iteration of testing took place.
2) Reconfiguring the test framework and the test library to support the functional and regression testing of the new **ATHEX FIX Server**.

The testing scope included functional and regression testing of functional changes in new ATHEX components. This case study reviews the functional aspects of the work that has been conducted as well as highlights the benefits of passive testing for verifying the quality of large-scale transaction processing systems. In particular, we concentrate on the advantages of using passive testing and historical test run data for detecting regression issues, monitoring system performance and health over time, detecting potential gaps in the systems' test coverage, among other use cases.

3. Trading System Migration – Functional and Regression Testing of ATHEX OASIS Platform

The **ATHEX FIX Server** is a new FIX trading interface of the ATHEX OASIS Trading platform enabling member firms to submit orders, quotes and/or trade capture reports and receive real-time information on executed trades. This interface is a point-to-point service based on the TCP/IP and FIX technology and industry standards. The session and application event models and messages are based on version 4.4 of the FIX protocol.

The legacy **ATHEX GW** – to be decommissioned after the exchange's migration to the **FIX Server** – is a component that previously needed to be installed on the client side to communicate with the matching engine via the COMMS Server, which contributed to the overall complexity of the client connectivity mechanism. For the migration period, both the COMMS Server and the ATHEX FIX Server were simultaneously connected to the AOM Matching Engine. The pre-migration configuration of the OASIS Platform [See Fig.1 below] provided ODL (Order Data Link, OASIS-native API) and FIX connectivity services.

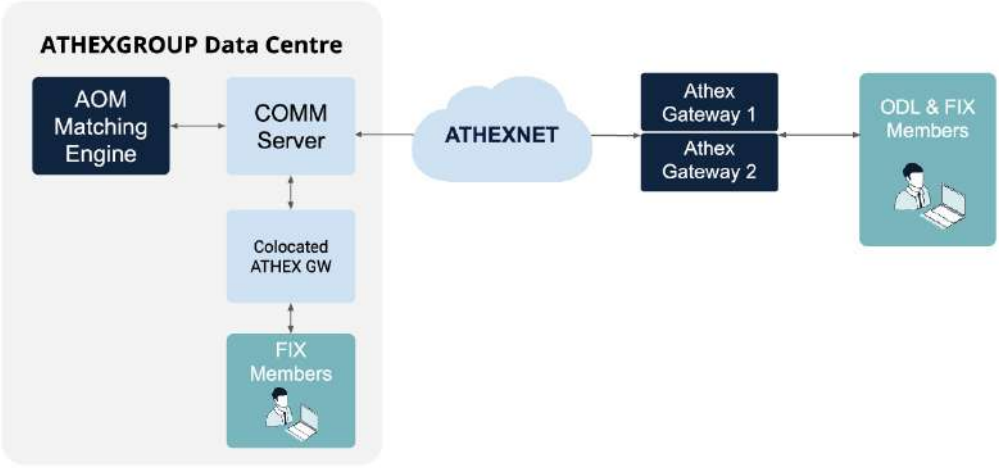


Fig.1 *Pre-migration configuration of client connections on the OASIS Platform.*

The test framework update required taking the following steps:
1) Bringing up to date the test library, reference data and connectivities checks for the ATHEX GW – 5600 functional test scenarios overall – making sure the library covers minor changes introduced over the previous two years.
2) Checking the validity of the updated test cases for the new FIX Server and adapting them to the new server-specific functionality.
3) Setting up a new 'Big Button' configuration for active functional testing. 'Big Button' refers to a fully automated testing workflow launched with a single click and performed on demand by the internal team after Exactpro hands over the respective configuration and instructions.

The new configuration [See Fig. 2] entailed:

1) Elimination of the ODL services;
2) Removal of the COMM Server;
3) Integration of the FIX Server;
4) Addition of DropCopy (DC) services;
5) Introduction of new fields in the FIX Server;
6) Introduction of new logic/elimination of outdated logic.

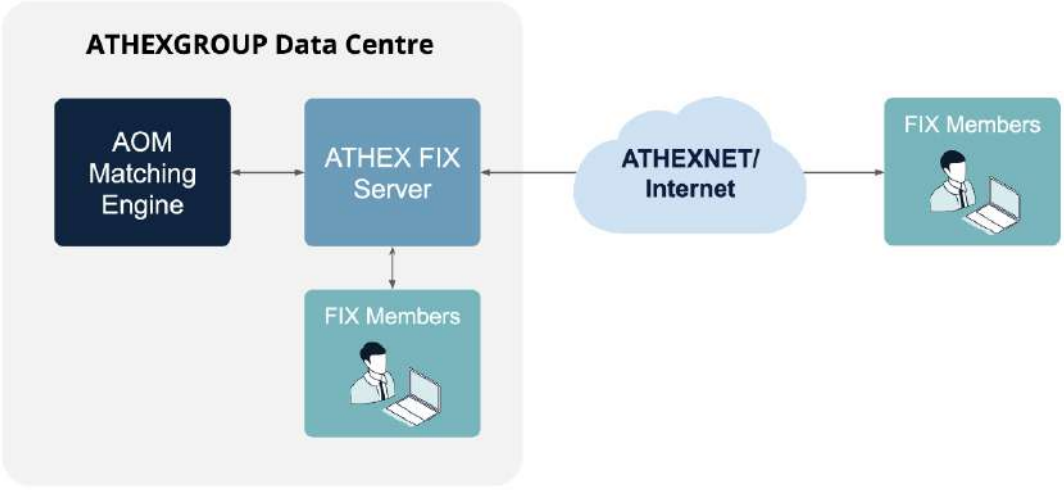


Fig. 2 *Post-migration configuration of client connections on the OASIS Platform.*

This brought forward the following respective testing tasks:

1) Checking for potential test coverage gaps occurring in the absence of the ODL services being eliminated. If gaps in the functionality coverage were to be detected, some of the ODL-specific functionality would have to be transferred from the ODL scripts to the new FIX services scripts. For example, off-book trading scripts were only covered by the ODL-services scripts and had to be recreated from scratch for FIX services.
2) Checking whether all the required script values, in particular, 'enumerated' values, are covered by testing activities, as expected.
3) Updating all test scripts, with the exception of the ODL scripts being eliminated – about 4,500 scripts overall – with new DropCopy verification checks.
4) Adding field checks to accommodate for the newly introduced fields of the FIX Server.
5) Accommodating for the new logic, including removing coverage of some legacy logic that became irrelevant.

The new configuration was also applied to the Big Button framework delivered previously. As a result of these steps, the Exactpro–ATHEX test framework was completely upgraded and reconciled with the current configuration of the OASIS platform.

An **end-to-end test library** is one of the deliverables provided by Exactpro after test completion. Having the test library on hand allows the Athex team to run regular maintenance and regression tests against their system in the Big Button mode.

Nikos Antonopoulos, Head, Trading Systems Development department, Athens Exchange Group says, "Exactpro has been a valuable partner to ATHEX for several years now. Throughout this partnership, we have collaborated to create a substantial and comprehensive test library that has been instrumental in automated regression testing across multiple versions, leading up to the introduction of ATHEX FIX. The investment in functional testing has yielded consistent results, effectively validating existing functionality and bridging significant gaps left by unit testing, manual testing and UAT. Once again, in this development cycle, Exactpro specialists have worked closely with our teams, contributing to the solidification of the product in all aspects, including functional and non-functional requirements, as well as documentation. The collaboration has built confidence levels, paving the way for a successful go-live.

"After a particularly demanding project with significant risk factors, ATHEX FIX is currently live marking a smooth and incident-free transition to production, without disrupting our daily operations. We are proud we have achieved the desired performance goals, with a nearly 50% reduction in round-trip time for ATHEX colocation clients. The tools and expertise provided by Exactpro have been instrumental in accomplishing these goals."

The test data received can also be further analysed with the help of Exactpro's th2 framework to leverage passive testing methods for future iterations of the system under test. The next section of our case study discusses the big data analysis features that th2 is equipped with to account for the growing complexity of global financial systems.

4. Passive Testing and Results Analysis Automation with th2 Data Services

Exactpro's th2 framework is equipped with a Data Services feature to address the issue of large amounts of data produced by the components of a transaction processing system. Data Services allow us to store all test run data in a unified way and conveniently visualise the test run statistics (from both messages and events):

- Data can be aggregated into respective stats – graphs/charts broken down by required metrics – for easy test report generation and comparison [see Fig. 3];
- Analysis can also provide information about the nature of inputs – permutations used in the tests, the number of messages, their types, and so on.

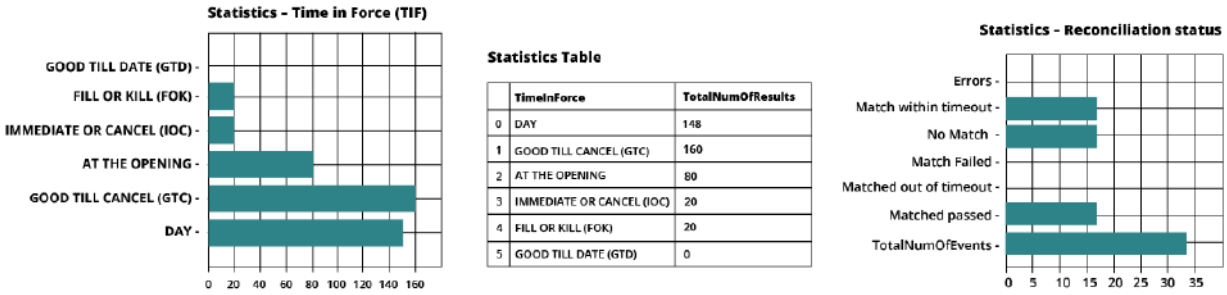


Fig. 3 Sample Data Services statistics.

Test run data retained from several system configurations can also be used for the benefit of the current system. Let's look at this passive testing use case in more detail.

4.1. Automating Passive Reconciliation with Data Services

For all test methods used, all test data generated as a result of the test tools' work is collected into the **Cassandra database**. Cassandra is a part of th2 framework's **Data Services** component that stores all messages and events of the test runs in raw format.

When data is parsed, it can be used by Recon – the reconciliation component responsible for the rule-based checking approach within the th2 framework [Fig. 4]. Reconciliation checks performed by Recon can be written in Python, Java, Kotlin or C++. Scripting reconciliation checks implies that specific rules are defined for matching the messages by a set of unique criteria.

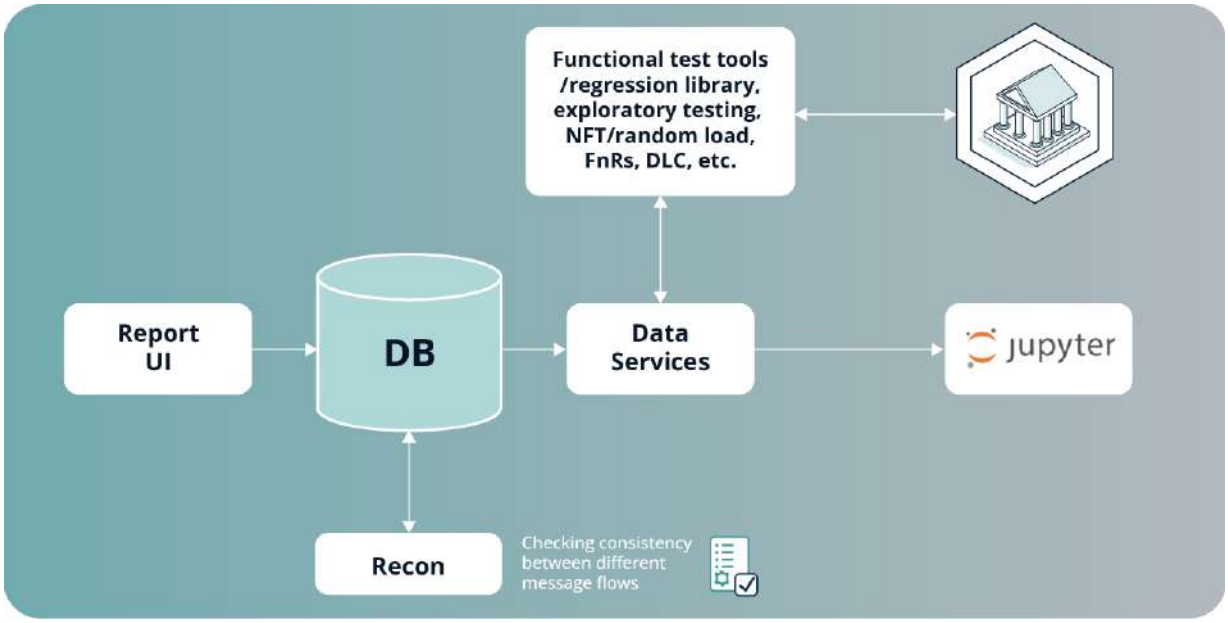


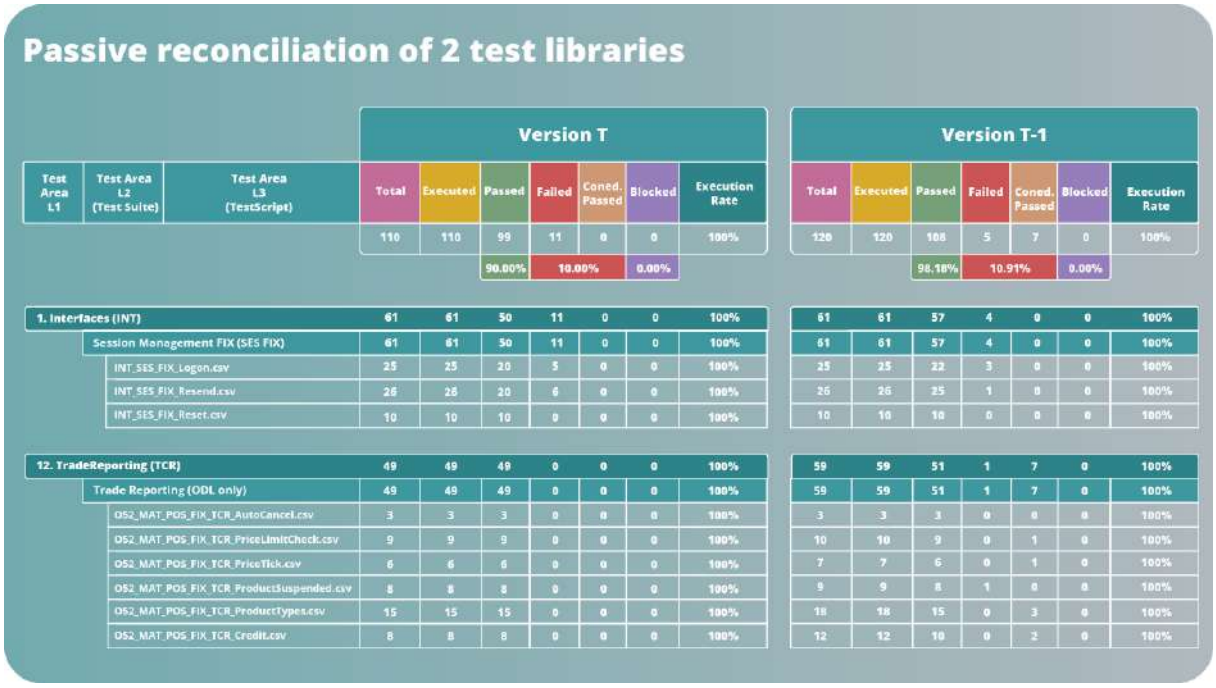
Fig. 4 The variety of inputs collected by Data Services and leveraged by Recon.

Keeping data consolidated allows us to compare two or more transaction data streams, in our case – the data of older versions of a test library with the data received from testing the current system configuration. Such a high-level comparison of 'passed' and 'failed' events is displayed in Fig. 5.

	Passed Actions	Failed Actions
OASIS 5.4 Test Execution	4096	252
Big Button OASIS 5.2 Test Execution	4108	85
OASIS 5.2 Test Execution (Legacy)	4142	46

Fig. 5 *Sample Data Services pie charts indicating numbers of 'passed' and 'failed' tests over several test runs.*

A more in-depth look at this data helps detect inconsistencies between the two data streams that require further investigation [Fig. 6].



Passive reconciliation of 2 test libraries

Test Area L1	Test Area L2 (Test Suite)	Test Area L3 (TestScript)	Version T Total	Executed	Passed	Failed	Coned. Passed	Blocked	Execution Rate	Version T-1 Total	Executed	Passed	Failed	Coned. Passed	Blocked	Execution Rate
			110	110	99	11	0	0	100%	120	120	108	5	7	0	100%
					90.00%	10.00%		0.00%				98.18%	10.91%		0.00%	
1. Interfaces (INT)			61	61	50	11	0	0	100%	61	61	57	4	0	0	100%
	Session Management FIX (SES FIX)		61	61	50	11	0	0	100%	61	61	57	4	0	0	100%
		INT_SES_FIX_Logon.csv	25	25	20	5	0	0	100%	25	25	22	3	0	0	100%
		INT_SES_FIX_Resend.csv	26	26	20	6	0	0	100%	26	26	25	1	0	0	100%
		INT_SES_FIX_Reset.csv	10	10	10	0	0	0	100%	10	10	10	0	0	0	100%
12. TradeReporting (TCR)			49	49	49	0	0	0	100%	59	59	51	1	7	0	100%
	Trade Reporting (ODL only)		49	49	49	0	0	0	100%	59	59	51	1	7	0	100%
		OS2_MAT_POS_FIX_TCR_AutoCancel.csv	3	3	3	0	0	0	100%	3	3	3	0	0	0	100%
		OS2_MAT_POS_FIX_TCR_PriceLimitCheck.csv	9	9	9	0	0	0	100%	10	10	9	0	1	0	100%
		OS2_MAT_POS_FIX_TCR_PriceTick.csv	6	6	6	0	0	0	100%	7	7	6	0	1	0	100%
		OS2_MAT_POS_FIX_TCR_ProductSuspended.csv	8	8	8	0	0	0	100%	9	9	8	1	0	0	100%
		OS2_MAT_POS_FIX_TCR_ProductTypes.csv	15	15	15	0	0	0	100%	18	18	15	0	3	0	100%
		OS2_MAT_POS_FIX_TCR_Credit.csv	8	8	8	0	0	0	100%	12	12	10	0	2	0	100%

Fig. 6 *High-level representation of passive reconciliation of two test libraries.*

By juxtaposing similar events and highlighting the disparities in the complex test libraries, Data Services can effectively help reveal potential regression issues or missing verification checks.

As outlined earlier, the statistics are further visualised and analysed by the testing team. The sample report in Fig. 7 reveals several field states whose verifications were missing from the test coverage as a result of the trading system migration (e.g. trade_cancel, trade_correct, and a few other execution types). The report is simplified for the purposes of the case study.

Long-term, tracking inconsistencies between a selection of test library runs allows us to reveal regression more effectively, detect floating issues and look into their root causes, as well as assess the impact of changes on system performance [see paragraph 6.3] over several system iterations.

	ExecType (150)	TotalNumOfResults
0	NEW	XXX
4	CANCELLED	0
5	REPLACED	XXX
8	REJECTED	XXX
9	SUSPENDED	0
C	EXPIRED	0
D	RESTATED	XXX
F	TRADE	0
G	TRADE_CORRECT	0
H	TRADE_CANCEL	0

***Fig. 7** Sample example of field verifications revealing test coverage gaps.*

5. Passive Testing Applications with th2 Data Services – Model-based Approach

The ability to demonstrate bi-directional connections between the requirements and the test products is an important project management and compliance aspect, as well as a criterion of system readiness. However, simply relying on a traceability matrix is not enough, as it does not guarantee the quality of the actual test coverage.

Software testing is context-dependent, thus, checking all the traceability matrix boxes is secondary to understanding how extensively a test library covers the entire functionality of the system. The only marker of robust test coverage is an end-to-end test library.

> "What a typical traceability matrix doesn't factor in is the fact that the specified requirements do not just exist in isolation: they need to be tested together, under load, as part of a model that describes the system as a whole," **comments Alexey Yershov, Head of the Global Exchanges Division, Exactpro**.
>
> "Special attention has to be paid to the vulnerabilities and errors that are likely to occur, for instance, when the system's functional and non-functional attributes interface. To reveal these, we need to make sure that every requirement is covered by multiple tests, many times, with alternating parameters and under various loads, rather than with a single test, as dictated by the simplistic model of the traceability matrix."

The source of true traceability is a generative model that produces an infinite number of tests, after which the necessary data can be extracted and analysed to reveal potential gaps. The approach is especially effective when test results produce a discrete set of parameter values in messages, such as in the case of protocol-based interactions between a financial system and its clients, where messages containing values of different types are transmitted via industry-standard or proprietary protocols.

The end-to-end approach to collecting system data enables us to see hidden dependencies and patterns that otherwise would have been overlooked. This becomes especially relevant when running high volumes of tests.

5.1 Data Services Implementation for FIX

The Data Services approach works for all financial protocols, but we will look at a widespread case – FIX, the Financial Information Exchange Protocol. Its fields are often limited by a set of values. Each message is a sequence of packages transmitted over TCP and containing tag-value pairs separated by pipes. Let's see an example:
8=FIX.4.4 | 9=420 | 35=D | 49=EP_Sender1 | 56=ATHEX | 34=18138 | 48=MOTO | 54=1 | 40=1 | 38=10000 | 59=1 | 44=0.62 | 11=479236360482EP | 10=109 |
The message is conveying that we are using the FIX 4.4 protocol and are requesting the creation of a new market order to buy ten thousand lots of MOTO shares at a price of 0.62 [Fig. 8].

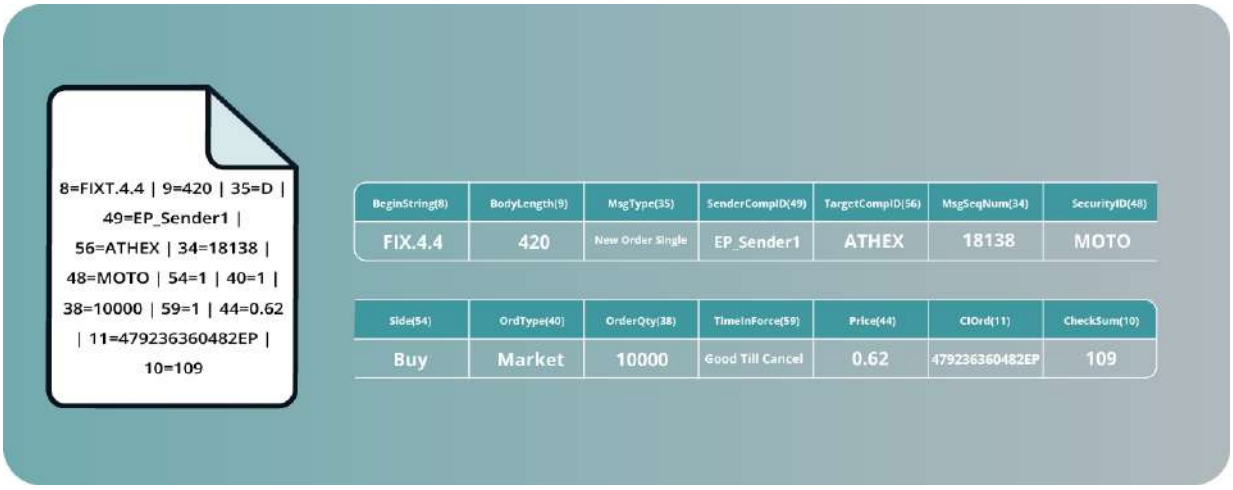


Fig. 8 *Sample FIX message.*

Within the limits of this single message, using just the two fields (tags) – 'Order Type' and 'Time in Force' – we can have X combinations which differ in behaviour. This would result in scores of such messages we can use in testing [Fig. 9].

ID	OrdType (40)	TimeInForce (59)	Total Num
1	Market	Day	XXX
2	Market	Fill Or Kill (FOK)	X
3	Market	Immediate Or Cancel (IOC)	XX
4	Limit or Better	Day	XXXX
5	At the Opening	Good Till Date (GTD)	X
...			

Fig. 9 *Sample permutations produced by the possible values of the 'Order Type' and 'Time in Force' fields.*

5.2 Machine-readable Specification

As previously mentioned, the Data Services feature allows th2 to collect messages in raw format, process them, and store them in Cassandra for further analysis. There are three main stages to the process [Fig. 10]:

Fig. 10 *Stages of data handling with th2 Data Services.*

1) Collecting all possible messages, their fields and values through network capture (TCP dump) or from the system logs;
2) Retrieving the necessary data from the logs or the TCP dump after the full run of the test library and storing the data in Cassandra;
3) Analysis and customised visualisation of the results from the database.

Our purpose is to save messages, their possible fields and values, as machine-readable. The process of collecting TCP dumps, creating databases and running the script to select values is fully automated using th2. It allows us to see customised by-value dynamics for various time windows and system versions. It is possible to sort the data by messages, fields, tags, as well as display unexpected values, etc.

Having executed these steps, we receive a certain representation of the test coverage, check its quality and reveal the unexpected system output values. The collection of these test run results essentially becomes an up-to-date machine-readable specification of the system under test.

Instead of establishing a limited number of scenarios for each requirement as our baseline (as would be the case with a typical traceability-matrix approach), we create a model based on the complete dataset acquired as a result of testing: functional and non-functional, manual and automated, regression and exploratory testing, both using fixed data and fuzz testing.

In addition to allowing us to compile customised Test Results Analysis reports, th2 Data Services are able to generate transition diagrams and latency charts, to provide a more holistic picture of the system's health. More customisation is available with Data Services upon request.

6. More Advanced Passive Testing Techniques

Apart from automatic creation of comprehensive test libraries, passive testing can also be effectively used for other business use cases. We will briefly recap two of them.

6.1 Conformance Certification

Conformance certification (also known as conformance testing) is a mandatory step in ensuring that customer systems conform to officially declared exchange/broker certification rules. Conformance certification is conducted in order to prevent compatibility issues between the trading platform and the trading participants' systems.

Some of the typical challenges faced during client conformance testing are world time zone disparities, high performance demand on the system due to varying customer demand, the necessity of deep domain expertise from the team conducting the certification, as well as ensuring comprehensive coverage.

Trading venues' certification teams use a number of approaches to perform certification testing procedures. All of them can reap the benefits of passive testing automation by using a scalable technology solution for conformance testing and the analysis of test results.
Please refer to our case study on Automating Customer Conformance Certification for an in-depth look.

6.2 Reconciliation Testing Under Load

As outlined in part 4.1, validating a network's performance under load is a vital part of the end-to-end testing approach. Provided that one's data capture mechanisms are reliable, network capture data leveraged via passive testing can be used for the system's performance monitoring and latency measurement. Inbound and outbound messages are matched, after which latency for every individual request is calculated. Exactpro's approach is to aggregate data as latency distribution charts, instead of averages or percentiles, to provide a better understanding of the system behaviour dynamics over particular timeframes.

Conducting reconciliation testing under load proves effective for detecting race conditions and other issues that occur at the confluence of functional and non-functional testing and cannot be detected with either one of them separately. The approach is a vital part of the verification of complex financial systems, by nature exhibiting non-deterministic behaviour (these include but are not limited to derivatives markets with implied liquidity).

Summary

In this case study, we have demonstrated the steps taken to upgrade the Exactpro–Athex Exchange testing framework to support the planned migration of the Athex OASIS trading system to the new FIX Server.

The project's testing scope included functional and regression testing of the changes introduced to the ATHEX trading system. This case study includes a review of the functional aspects of the conducted work and highlights a broad range of benefits of passive testing for verifying the quality

of large-scale transaction processing systems. We have highlighted the tools that enable the th2 framework to implement advanced data analysis techniques on the data collected during passive testing.

The passive testing benefits discussed include but are not limited to creating comprehensive test libraries, automating regression testing and passive reconciliation checks, monitoring the system's performance and health over time, detecting floating issues and issues manifesting themselves under unique conditions, such as when the system is stressed, which is a crucial type of validation for high-availability mission-critical financial platforms.

Last but not least, the data collected as the result of passive testing provides a straightforward way of generating audit trails for compliance purposes. It also presents a self-contained tool for improving the test coverage of the system.