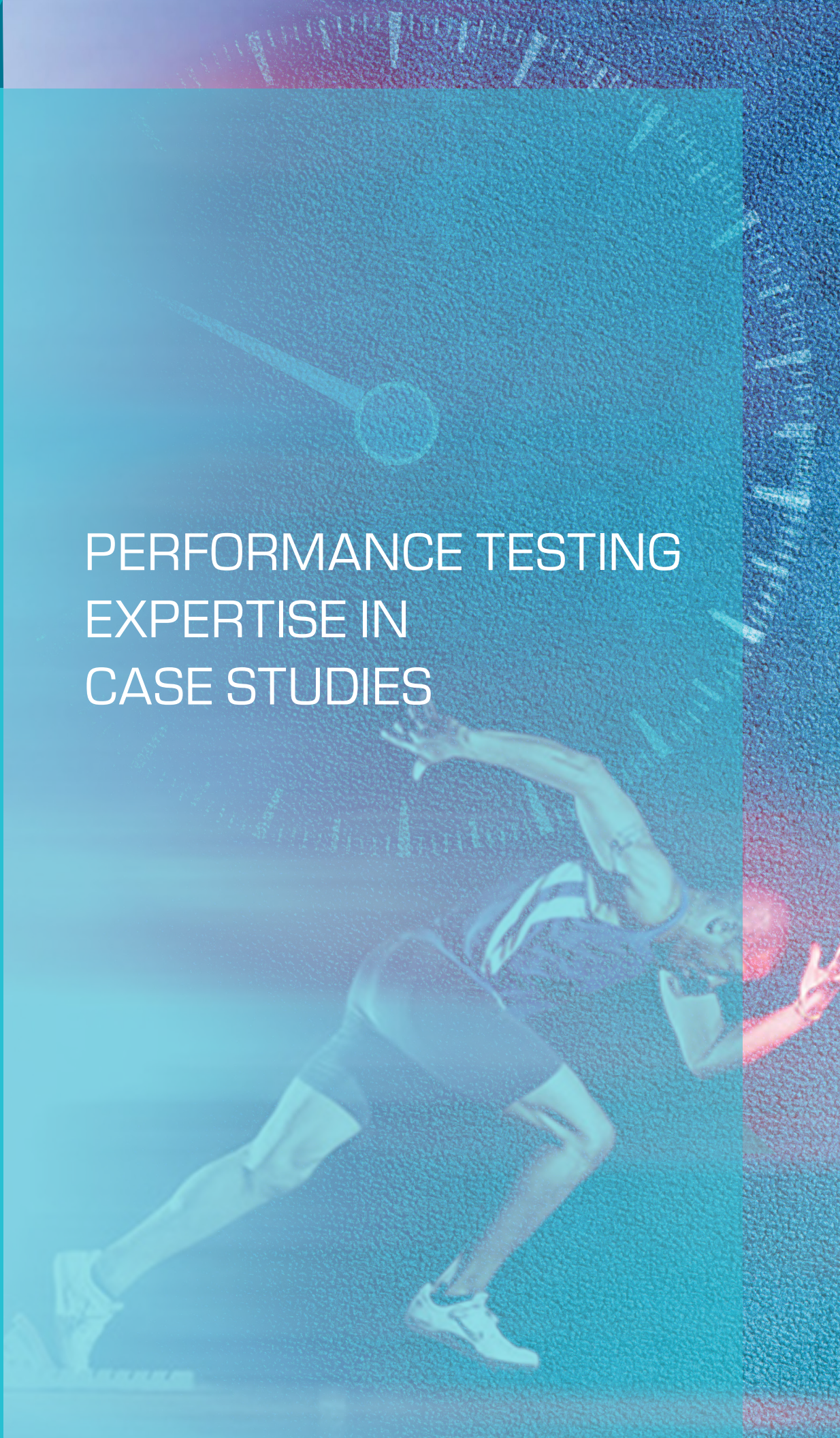
eleks® TESTING & QA

PERFORMANCE TESTING EXPERTISE IN CASE STUDIES



Case 1

Solution:

ELEKS' team has developed a custom test framework designed to accommodate numerous types of testing and measurements:

- Accuracy testing – during these tests the actual image recognition capabilities of the product were tested
- Latency measurement – these checks allowed the team to ensure that search of a certain pattern complies with the defined time scope
- Through-put measurement - these checks allowed the team to ensure that system is able to process required amount of requests per unit of time
- Load testing – was utilized to ensure that the system behaves according to requirements under projected query load
- Stress testing – helped to reveal the bottlenecks in the code and the "ceiling" of the hardware capabilities of the environment.

The test framework has been built using Python. It provided QA team with an easy way to create test-cases and scenarios, without requiring in-depth Python skills from all members of the team. The system performance was measured using Nagios– a popular open-source monitoring system deployed across all environments. Test results were visualized via a web-site built using Python: a number of detailed reports about system accuracy and its performance were generated automatically after each test execution.

Customer
High-tech image-search startup from California, USA

Challenge:

1. High-performance image search engine with requirement to run search through millions of images in less than a second.
2. Distributed and heterogeneous nature of the system: dozens of servers hosting software written in C++, Java, PHP and Python.
3. Continuous delivery over multiple environments (development, staging and production).

Result:

Test framework helped:

- Continuously measure system accuracy
- Determine and eliminate performance bottlenecks
- Continuously ensure system integrity, and prevent regressions

Technologies & tools used: C++, CUDA, OpenCL, Java EE, PHP, Python, MySQL, Mongo DB, Nagios.

Case 2

During automated testing Windows Performance Toolkit (WPT) was chosen to collect the application statistics of OS boot sequence during system load. ELEKS provided a reasonable OS coverage in context of configuration. In order to minimize the efforts for operating system image deployment the automated scripts were created. Scripts were written using AutoIt tool in conjunction with utilization of such technologies as CMD and PowerShell.

In manual testing, one of the factors that influenced the result matrix was system operability and usability. It was up to software testers to provide feedback on overall product performance apart from pure statistic data. After several cycles of checks the gathered statistics and feedback were compiled in a report matrix. Due to the fact that customer's product is highly secured, the report was sent to the customer for further analysis.

Result:

Performance testing helped:
Define the critical issues of the product
Concentrate on code optimization
Enhance the overall system performance

Customer
A leading provider of security software to millions of users across the globe.

Challenge:

1. Continuous need for Performance Testing of each build produced
2. Requirement to check how committed changes influenced the overall system performance

Solution:
After analysis of the customer's requirements the Performance Testing was chosen as the best approach that could meet the needs and scope of the project.

ELEKS' team performed the following tasks:

1. Requirement analysis;
2. Automation scripts development (using AutoIt, batch scripting and PowerShell);
3. Clean environment deployment using scripts;
4. Both manual and automated Performance Testing approaches (using Windows Performance Toolkit (WPT));
5. Results analysis using gathered statistics.

Technologies & tools used: AutoIt, PowerShell, CMD, Windows Performance Toolkit

Case 3

Solution:

ELEKS' team performed the following tasks:

1. Created two very similar environments with two product versions. The environments had to be identical and independent from external influence (e.g. performance execution results had to be the same in different period of time), therefore, we chose private cloud system VMWare vSphere. Virtual client for Performance Testing was deployed on this cloud.
2. Defined the most critical part of application from the performance perspective, and created over 30 use-cases which were automated with jMeter.
3. Performed numerous test case executions and collected performance metrics for analysis.
4. Performed metrics analysis. Calculated:
    - Average response time
    - Minimum and maximum response time
    - Standard deviation
    - Through put, etc.

Customer

A Germany-based company that provides market leading Training Management System for aviation organizations.

Challenge:

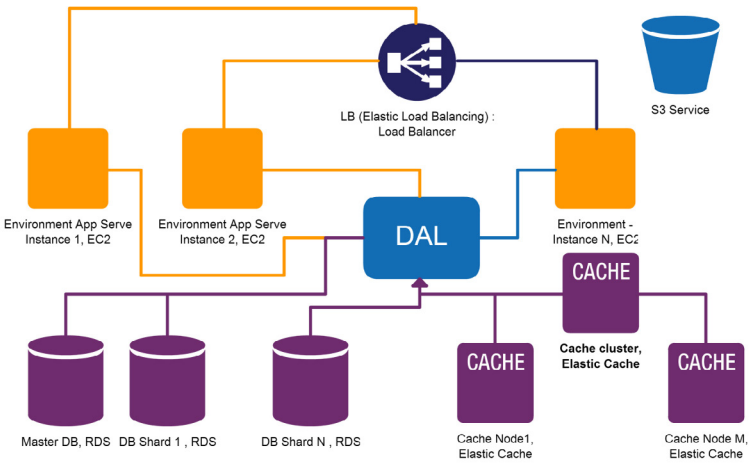
The customer's product was on the market for two years and deployed to many customers. When the new version was released the customer wanted to test it to see how the new version correlated with the previous one, whether it was not too slow, stored procedures, did report generation, etc.

Result:

Compared metrics for both product versions, as well as version assessment of customer's solution, which helped to define the performance of the new product version.

Technologies & tools used: jMeter, vSphere, eViews.

Case 4



Solution:

ELEKS' team performed the following tasks:

1. Created demo environment for performance testing on AWS
2. Defined use-cases and created automated scripts
3. Performed initial Load Testing to collect rudimentary metrics
4. Altered AWS components configuration to find the most effective one
5. Performed Stress Testing with load step starting from 1000 virtual users and up to 15 000 users, in order to test proper reaction of elastic load balance and elastic cache.

Customer

A US-based company that offers a web portal for tattoo designers and users.

Challenge:

The customer asked ELEKS to develop and test (including Performance Testing) its product which was expected to have 15 million users per month. The application was deployed on AWS cloud.

The main challenges were:

- Define proper settings for elastic load balance
- Define proper settings for elastic cache
- Define type for EC2 instance
- Define the most cost-effective configuration
- Locate critical issues and bottlenecks in application architecture.

Result:

1. Defined proper settings for elastic load balance and elastic cache
2. Defined the most cost-effective configuration
3. Figured out application architecture issues and bottlenecks.

Technologies & tools used: jMeter, AWS

Case 5

Solution 1:

1. Creatde a copy of the front-end and back-end production environment on the vSphere server for testing improvements.
2. Created virtual environment for running load, stress, and performance tests
3. Defined and automated test-cases for web-site and mobile API
4. Performed initial tests and collected metrics:
    - Average / Minimum / Maximum Response Time
    - Time to firs byte
    - Delivered Load
    - Response Times Quintiles
    - Response Times Distribution
    - Server CPU usage
    - Server Memory usage
    - hSQL queue duration
5. Performed metric analysis

Solution 2:

1. Automated the process of loading the virtual environment and tests execution on it

2. Set up automatic load balancing control for front-end servers
3. Created a system for notification in case of problems with the production environment during Load tests
4. Automated the process of saving the tests metrics

Customer

A well-known Dutch online supermarket that offers product delivery to customers' homes.

Challenge:

1. Increase web-site and mobile API performance
2. Setup the system and develop automation scripts to run the load test after each update on production server

Result:

Two times increased system performance.

Technologies & tools used: Jmeter, vSphere, vSphere PowerCLI, PowerShell, ANTS Performance Profiler, DotTrace, SQL Server Profiler, NewRelic, loadosophia.

Case 6

Customer
A leading provider of time tracking system and cost recovery system.

Challenge:

1. Significant system slowness was observed when it was going through the process of rollout to client production environment. This required to create an environment for system testing – perform Stress Testing.

Solution:

ELEKS' team performed the following tasks:

1. Created an analogue of client-side application which allowed to set up needed levels of workload;
2. On server-side there were added performance-tracking counters;
3. Prepared cloud-based environment for client-side application in order to emulate required workload;
4. Used vSphere SDK to create centralized management system for virtual machine batch image cloning;
5. Done remote configuration of virtual machine environment, including client application deployment and logging system preparation;
6. Scheduled workload scenarios of execution and automated logs collection;
7. Used pre-defined post-processors.

Result:

1. Identified system's gaps and bottlenecks
2. Fixed system slowness on server-side
3. Optimized data storage procedure
4. Decreased hardware system degradation
5. Specified minimum of hardware and software requirements configuration.

Technologies & tools used: C#, .NET, vSphere SDK, PowerShell.

Contact Us:

ELEKS Headquarters

Eleks, Ltd.
7 Naukova St., Building G
Lviv 79060, Ukraine
phone: +380 32 297-1251
fax: +380 32 244-7002

US Office

Eleks, Inc.
701 North Green Valley Pkwy
Suite 200
Henderson, NV 89074
phone: +1 866 588-0113 (toll-free US)
+1 617 600-4059
fax: +1 678 905-9508

UK Office

ELEKS Software UK, Ltd.
5 Harbour Exchange
South Quay
London, E14 9GE
phone: +44 203 318-1274

email: eleksinfo@eleks.com
www.eleks.com