A leading analytical instrument’s manufacturer wanted to have an automated quality control system in place with the following intended benefits.

- Fully automated quality control system
- Maximum coverage of testing
- Testing without the real hardware
- Multilingual testing
- Multiplatform testing
- Daily testing

The solution was expected to control the quality of product by continuous verification process. Optimization of testing effort and wider test coverage were the other objectives.

NeST has proposed its proven CIVIM (Continuous Integration, Virtualization, Internationalization and multiplatform support) quality control framework for the customer. The framework was customized to cater the requirements of the project and implemented under the guidance of test automation experts. A highly configurable hardware simulator was implemented to cover the test scenarios that interact with hardware.
About the Customer

A leading manufacturer in analytical and industrial technology with worldwide operations based in US, Japan, Europe and China. They are mainly specialized in fields of x-ray diffraction (XRD), thin film analysis, x-ray fluorescence spectrometry, protein and small molecule crystallography, semiconductor metrology, computed tomography etc.

Business Challenges

The product supports three languages (English, Japanese, and German) and two operating systems (Windows 7 and Windows 8). The processor architectures like 32 bit and 64 bit are also supported by the product. To ensure the coverage, 700 test cases had to be executed against 12 platform configurations. Manual testing of the product against all platforms configuration takes about 240 PD.

The Solution

NeST has proposed and implemented CIVIM quality control framework for the project. CIVIM integrates different concepts like Continuous Integration (CI), Virtualization, Internationalization and Multiplatform Support, which ensures health check of product at regular intervals.

The architecture of the solution consisted of five major blocks: Continuous Integration Server (CI Server), Configuration Management Server (CM Server), Build Server, Backup Server, and Test Server. The tasks to be performed were distributed into different servers so that the load was distributed evenly and total time of execution could be optimized. Each server has got its on responsibility of performing defined tasks and maintenance could be done independently.

All the activities related to automated testing were scheduled to happen nightly.

At the scheduled time, CI Server initiates the task for downloading the source code and build scripts from CM Server to Build Server. The source code of the product is then built and installation setup is created for successful builds. ‘Daily installation setups’ are kept in a store for future reference.

The build server performs the static code analysis. It measures the code complexity, code coverage and run static analysis tools. The reports of the static analysis are generated, consolidated and sent to the CI server. The CI server publishes the
result in the web dashboard and also sends this to all stakeholders.

Once the installation setup is successfully created, CI Server sets up virtual machines corresponding to each test bed configuration in the Test Server after taking virtual machine images from the Backup Server. Now all the test machines, in the form of virtual machines, with required configurations are ready in the Test Server.

CI server accesses the product installation setup from build server and installs it in virtual machines in parallel. Test scripts are then downloaded from the CM Server into each virtual machine and tests are executed in parallel. Test reports corresponding to each configuration are generated in the corresponding virtual machine.

The test reports are consolidated and published in a web dashboard. They are also sent to the project team. The virtual machines are then switched off.

<table>
<thead>
<tr>
<th>#</th>
<th>Tool /Script</th>
<th>Propose</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Teamcity</td>
<td>Automated Continuous Integration</td>
</tr>
<tr>
<td>2</td>
<td>Powershell</td>
<td>Scripting</td>
</tr>
<tr>
<td>3</td>
<td>Visual Studio 2012</td>
<td>Building</td>
</tr>
<tr>
<td>4</td>
<td>NSIS</td>
<td>Installation</td>
</tr>
<tr>
<td>5</td>
<td>Resharper</td>
<td>Code analysis</td>
</tr>
<tr>
<td>6</td>
<td>Hyper-V</td>
<td>Virtualization</td>
</tr>
<tr>
<td>7</td>
<td>PsExec</td>
<td>Remote commands execution</td>
</tr>
<tr>
<td>8</td>
<td>TestComplete</td>
<td>Automated Testing</td>
</tr>
</tbody>
</table>

**Table 1: Tools and technologies used**

**Business Impact**

A versatile quality control framework has been set up. Nightly testing of the product against the latest source code check-ins was accomplished. 95% of the test cases could be automated and testing against all target platforms was ensured. Defects could be revealed at the earliest since nightly testing was enforced. Product verification was done effectively in each sprint of agile development. Test execution without real hardware was achieved and Return on Investment was achieved by the end of first year itself.

**Benefits**

1. Greater Return on investment (ROI)
   Test Automation using CIVIM helps in enhancing the test execution speed at a lower cost and higher ROI. CIVIM framework runs continuously at a faster
test execution speed which increases the productivity of the organization.

2. Unattended testing
   CIVIM enables unattended testing of the entire software.

3. Well formatted result logs
   A well formatted test report is available. This helps in easy analysis of the test results which helps in reducing the effort spent on test result analysis and ensures readability of test logs.

4. Early stage defect detection
   Since the framework performs QA check every night, induced defects are caught in the very next day itself.

5. Better Infrastructure Utilization
   Since the test automation is scheduled to be carried out at night, it helps in the most effective utilization of the infrastructure.

6. Increase over all product quality
   CIVIM ensures an increase in the overall product quality. It ensures complete and continuous test coverage at a higher test execution rate. This helps in testing more or complete functionalities of the product as compared to the manual testing. The identification and isolation of the defects can be increased markedly, resulting to increased product quality.

7. Evaluation of the quality of product development
   The framework provides the history of test reports on a web dashboard. This helps to understand whether the product quality is enhanced or not based on accumulated bug trend.

8. Efficient regression testing on multiple platforms
   CIVIM supports efficient and effective execution of the regression test suite on multiple platforms without consuming time in project schedule.

9. Efficient regression testing of different language and regional settings
   The framework supports testing of different language and regional settings of the software product.

10. Detailed code coverage and static code analysis report
    The static code analysis tool and code coverage tools integrated with the CIVIM framework enables the developers to get detailed report on code quality at an earlier stage.