Automated NFR Testing – Detect performance issues at early stage of product life cycle

Lanke Vishal[*], Pawar Atul[**]

[*] Siemens Technology Services Private Limited

Abstract- *Poor product performance is one of the major causes for its market failure. They tend to have their roots in inadequately drafted Non-Functional Requirements (NFRs), poor architectural and design decisions, and inadequate NFR testing practices right from the beginning. Performance, reliability, and availability considerations must be built into the system from the time it is conceived.*
Performance considerations and requirements are often not taken into account until the product is ready for system testing. "Build it, and then tune it" is often followed by organizations, which lead to product failure in testing scenarios.
In this paper we will be talking about experiences that we had faced in performance testing of life safety applications and devices, how we overcame challenges to improve product quality.

Index Terms- Device Loader, Device Simulators, PERFMON, and Report Generation using UNIX

I. Introduction

We are developing life safety emergency notification system that will protect, alert and inform people in case of natural calamities like Fire, Flood, Tsunami, etc. and in case of antisocial activities like Terror Attacks, etc. However, conducting performance test for this product was a challenge due to its complexity, simulating integrated devices, scaling up the number of the devices & generating user-friendly reports.

II. TEAM STRUCTURE

Product teams are split across different geographical locations. Team1 is responsible for NFR testing, manual testing, backend development. Team2 is responsible for product management, system architect and test architect. Team3 is responsible for the development & testing of user interface. Team4 is responsible for backend logic development.

III. CHALLENGES

Nature of product: We are developing & testing a "Life Safety Application" which should work on the worst day of customer. This meant that apart from all the functional requirements, non-functional requirements such as response time, scalability & reliability of the application should meet the expectations from an end user.

Expertise in NFR: Being the first release of the product, there was no prior data that would be set as a benchmark. Importantly test team too was very new to the non-functional testing & tools that may be required to develop the framework.

Geographically distributed team: We were facing problems of poor communication, specifically long feedback loops when teams were trying to understand performance requirements and get any questions or clarifications addressed. This had contributed to a lack of commitment and feeling of separateness between team members.

Delivery of devices delayed: vendor drove Delivery of devices that are integrated with the application. These devices were very limited in numbers [1 to 3 in quantity] & shared across development team, manual testing team, functional automation team and NFR team.

Performance monitoring: Application was in distributed network, so monitoring the performance of each server was challenging. In addition, stakeholders found it hard to interpret it at one go.

IV. RESULT OF CHALLANGES

With the challenges we had, there was delay in the start of NFR test execution. Execution started towards end of the product life cycle & as a result, the performance related issues were uncovered very late. This resulted into delay of milestone and shipment to the market.

V. LEARNING FROM PAST MISTAKES

We had a very complex application wherein hardware/devices were closely integrated with the software. Performance testing of such framework would need seamless flow of the data & way to measure its output. We primarily tried to focus on couple of areas to achieve this -

1. **Streamlining NFR Strategy:**

- NFR test cases were designed & mapped to the business requirement.
- Defined the guidelines for test execution [Ex. Delete all the dependent data from previous test case execution etc.]
- Decided to start the NFR testing early in the phase when manual test team starts its execution for a new build.
- NFR framework was automated.

2. **Automating NFR Framework:**

- **Identify Test Cases** - Test cases were identified for daily, overnight, weekly, and customer profile. Daily test cases run for 2HRS, overnight test cases runs for 8HRS, weekly runs for 24HRS. Customer profile test cases contain the set up identical to what customers use.
- **Create Test Data** - Manual test team creates analysis documents after reviewing feature specification. These analysis documents are used to create the test cases. Automation test team to create the test data then uses these test cases. In turn the test data is further used by NFR framework.That way we use uniform test analysis document & pre-requisite test data for all the testing teams.
- **NFR Test Set Up** – We have used windows backup and restore feature to create multiple versions of windows and system under test within 30 minutes. Hyper-V-Terminal embedded in windows is used to create and manage multiple virtual machines. We benchmarked a system using Microsoft Hyper-V-Terminal guide, internal expertise of the team members and guides that we received with life safety devices to make it reliable.
- **Tool Development** – Used to automate NFR test set up, test case generation, test case execution, 24*7 performance monitoring, result analysis and to find out performance state of the product
- **Reusable modules** - Modules were implemented in such a way that all teams with could use it minimal or no changes. Reusable modules consist of commands, logic, validation and test data, which can be customized for any product.
- **Device Simulators** - We have implemented infrastructure that can be used to simulate any device irrespective of its protocol. It simulates actual network traffic. A centralised service sends commands to any systems to simulate network level protocol. Simulated data is stored in central database. It is achieved without purchasing any hardware or commercial software. It uses test data, which is already reviewed. For example, if a driver is sending TCP command, simulator will act as TCP listener.
- **UI Automation** - We use existing function libraries of QTP that was developed by functional automation team to measure UI response of various UI activities. It is executed 24*7 without any manual interruption.
- **Automate End-to-End Test Execution** - Internal website is developed to select NFR server & start/stop/pause any performance test with appropriate status such as pass or fail. User can also drill down the test case to verify where it has failed.
- **Automated Report Generation** – We use UNIX to interact with windows performance monitor. It automatically generates performance report containing test case details and detailed analysis of performance issues in Cores, Memory, Disk, network, Process and 250+ parameters like dot net, SQL activities etc. It can be used in other products irrespective of product domain and technology. We took lab measurements at short intervals (every 15 sec) to enable detection of transient faults or progression to a fault condition.

For example, below report is automatically generated -

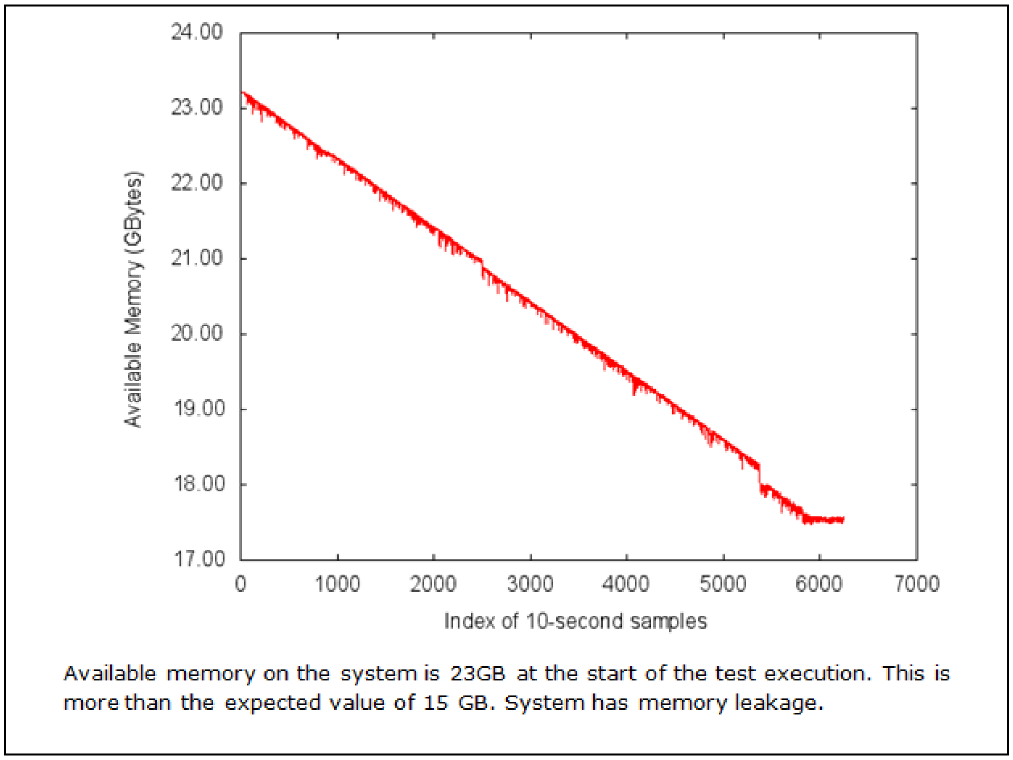

Figure 1: Automated performance report

3. **NFR Status Reporting**: We started providing daily status to the stakeholders that helped them understand the quality of the product on day-to-day basis.

VI. RESULTS

With the past learning's, we kept improving on the newly defined strategy. Below are the accomplishments/milestones that we achieved in the process -

- Time spent to implement automated test infrastructure – 6 months
- Generic framework developed in a way that can be customized for any product/application.
- Time to set up NFR server with thousands of users and devices - 20 Minutes
- Time to set up and start simulators for all the integrated devices [#17] – 15 Minutes
- Time to set up automated NFR report generation – 5 minutes
- Time to generate automated report after test case execution – 15 minutes

I. RECOMMENDATIONS

Performance test results are valuable assets because -

- Performance benchmarks can be used for comparison with previous and subsequent releases as well as with the competitor products if any.
- They tell us actual status on the way development is carried out.
- Can form inputs to configuration and load generation tools
- To help sales engineers confidently produce competitive offers in a timely manner

When performance requirements are being formulated - Performance bottlenecks are best identified in the early phase. Ensure that performance and scalability requirements are written in verifiable, measurable terms, and that they are linked to business and engineering needs.

At the architectural stage - Advice on the impacts of technology and design choices on performance, design test cases, set guidelines, implement necessary tools, plan performance tests that verify performance and scalability requirements are met.

Test execution – The closer your test cases, test data & test environments are in terms of hardware, software configurations to real world scenarios, the greater is the likelihood that your production release will not encounter too many performance issues.
Test Reporting - Provide clear picture to the stakeholders on test case execution's progress. Prepare reports and interpretations of the results that help the other stakeholders to make decisions about potential performance improvements.
At all stages during NFR testing - Act as a performance advocate for the project, ensuring that performance needs are addressed early.

APPENDIX

Table 1: Acronym

Acronym	Meaning	
NFR	Non-Functional Requirements - In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours.	
	Source	http://en.wikipedia.org/wiki/Non-functional_requirement
VM	Virtual Machine - A virtual machine (VM) is a software implementation of a machine (for example, a computer) that executes programs like a physical machine. Virtual machines are separated into two major classifications, based on their use and degree of correspondence to any real machine	
	Source	http://www.en.wikipedia.org/wiki/Virtual_machine
QTP	Quick Test Professional – HP QTP, an automated functional testing tool that helps testers to perform automated regression testing in order to identify any gaps, errors/defects in contrary to the actual/desired results of the application under test.	
	Source	http://www.tutorialspoint.com/qtp/

.

REFERENCES

[1] Virtual Machine - http://www.en.wikipedia.org/wiki/Virtual_machine
[2] NFR - http://en.wikipedia.org/wiki/Non-functional_requirement
[3] QTP - http://www.tutorialspoint.com/qtp/

AUTHORS

First Author – Lanke Vishal,MCA, NFR Test Leader, vishal.lanke@siemens.com, 9764748228
Second Author – Pawar Atul, BE (Chemical), Test Manager, atuldpawar@siemens.com, 9822441545