Warfighter's Mission-Critical System: Automated Testing and Test Management

H. Ferhan Kilical, Ph.D.

Technical Fellow, Electronic Systems

(Test , Test Automation, SOA, Performances Test

and Agile Methodologies)

Report Documentation Page		Form Approved OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **APR 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**
4. TITLE AND SUBTITLE **Warfighter's Mission-Critical System: Automated Testing and Test Management**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Northrop Grumman Electronic Systems,1580-A West Nursery Road,Linthicum,MD,21090**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

14. ABSTRACT
Warfighter?s mission-critical system: Agile automated testing and test management Do any of these problems sound familiar? ? Too many failed scripts ? Slipping schedules ? Automation tools that never get off the shelf ? Incomplete test coverage ? Do more with less, and ? Be creative in performing the business Then, come and listen to the story of Warfighter?s mission critical application testing. This case study reveals how testers used automated functional, performance, and Service Test scripts for a major mission-critical application that had to meet the most rigorous quality standards. Also, working in an agile environment, the team managed end-to-end requirements and defects and performed functional, SOA and performance testing. With risk based and automated test strategies, the team was able to do automated regression and smoke tests in a fast-paced development ? agile environment and managed test results, test sets, and test-related artifacts. It was a very successful project, completed on time with very limited resources.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT **Same as Report (SAR)**	18. NUMBER OF PAGES **33**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Abstract

Warfighter's mission-critical system: Agile automated testing and test management

Do any of these problems sound familiar?
  - Too many failed scripts
  - Slipping schedules
  - Automation tools that never get off the shelf
  - Incomplete test coverage
  - Do more with less, and
  - Be creative in performing the business

Then, come and listen to the story of Warfighter's mission critical application testing.

This case study reveals how testers used automated functional, performance, and Service Test scripts for a major mission-critical application that had to meet the most rigorous quality standards.

Also, working in an agile environment, the team managed end-to-end requirements and defects and performed functional, SOA and performance testing.

With risk based and automated test strategies, the team was able to do automated regression and smoke tests in a fast-paced development – agile environment and managed test results, test sets, and test-related artifacts. It was a very successful project, completed on time with very limited resources.

Acknowledgements

- Special thanks to the project team at Mission Systems, NG, Herndon.

Complex, Mission Critical Applications & Testing

WarFighter's Needs

One NET

Decision Tools

Joint Health Service

Joint Logistics Service

Joint WarFighting

Force Preparedness

Plan, Prioritize Data

Use Data,

See Data,

Capture Data

Anywhere, Any user

Application
Battle Command
Common Services

TIE DOWN
KMM
KVM / Ether Switch

Joint Operations

Location, Capture Data

- **Keeping Track of:**
  - Requirements
  - Defects
  - Test cases, test processes and test plan

- **Managing Testing Cycles**
  - Problems in infrastructure and scalability
  - Ongoing implementations and significant development initiatives
  - Significant degree of customization and integration
  - Limited resource availability
  - Integration with Complex Portal and Identity Management Solutions

- **Weak Testing Methodology**
  - Manually intensive
  - People - Not process driven
  - No automated testing capabilities
  - Not trusting anyone else to test

Testing Story

- When a major government contractor delivers software, that software must comply with the most rigorous quality standards

- By enabling both automation and proper test management, we benefit from critical advantages not offered by manual testing

- If in "agile" development model: Development cycles are short and tests are conducted at the same time as the coding

- Things can get particularly complicated when the team is testing SOA applications and performance-scalability testing

Waterfall Model

Developer
SA
IDM
Tester
Scrum Mast
Gov
Secu Rep
DB Rep

Developer
SA
WB/KMS
Tester
Scrum Mast
Govt Rep
Secu Rep
DB Rep

Developer
SA
InfraStruc
Tester
Scrum Mast
Govt Rep
Secu Rep
DB Rep

Developer
SA
Mapping
Tester
Scrum Mast
Govt Rep
Secu Rep
DB Rep

Developer
SA
Rep Analysis
Tester
Scrum Mast
Govt Rep
Secu Rep
DB Rep

Developer
SA
Reporting
Tester
Scrum Mast
Govt Rep
Secu Rep
DB Rep

Updated Requirements
Verify

Requirements
Verify

Specification
Verify

Design
Verify

Implementation
Test

Integration
Test

Development
Maintenance

Product Deliverable

Maintenance

Agile Model

Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5
Analysis	Analysis	Analysis	Analysis	Analysis
Coding	Coding	Coding	Coding	Coding
Testing	Testing	Testing	Testing	Testing

We Did Not Have Processes and Tools in Place

- The client had Rational tools (Req Pro, ClearQuest and Test Manager) but they were not properly used.

- And a lot of Excel sheets and Word docs. Never ending story of not being able to control requirements and defects …

- We bought Quality Center, and we made Rational tools and Quality Center tools talk to one another. We faced challenges with Firewalls and security.

- Meanwhile, we started working on the processes with the client and our NG internal processes for test.

- We started putting together IMS, built partnership with the client .

Agile Sprint Testing Workflow

Requirements provided by DISA

NG Team Develops Use Case

Agile Team Develops User Story

Tester Develops Test Cases Based on Input from Agile Team Members

Agile Team Accepts Test Case

Test Case Completed

Test Case Placed in TEAM/Quality Center

We had to make Rational Test manager and Quality Center Test Plan talk to one another.

Smoke Testing — Fails

Patch Testing — Fails

Regression Testing — Fails

CFS Smoke Test — Fails

Release Manager Assesses Impact

Test Cases Need to Be Modified *

Test Cases Placed in Rational Test Manager, DISA

Test Results Compiled and Submitted to Test Manager, Release Manager and CM

DISA Testing

Smoke Testing — Fails

Patch Testing — Fails

Regression Testing — Fails

CFS Smoke Test — Fails

Test Results Delivered to DISA

* In Agile Development, test cases are based on the input from Use Case, User Story and Agile Team. As they complete the build, it it tested and final test cases are formed

Automating Test Scripts

- Functional Testing

- Performance, Load, Stress Testing

- Service Test

- Security and SA Type of Testing

- Important Features in Relation to Automation:

  - Test Case, Requirement correlation

  - Defects, Defect Management and its correlation to requirements and test cases if possible

  - Test Scheduling IMS and test schedules and customer testing

Purpose of Automated Testing

- Checks virtually any functionality in application.

- Provides consistently re-usable tests to be run each time a site or application changes

- Shortens testing time especially regression testing.

- Tracks all test runs, logs, test results in one, shared repository.

Major benefits are:

- Reusability

- Consistency

- Productivity

- Team work environment

- Tests that need to be run for every build, sometimes referred to as Sanity tests (Smoke and Regression tests).

- Tests that use multiple data values for the same actions are known as Data-Driven tests (Equals to, =>, <=).

- Identical tests that need to be executed using different browsers (We are using IE6,7 and FF).

- Mission critical pages (Certain pages need to be checked all the time).

Agile Testing Cycles

- **Sprints Testing:** For every sprint, test team will have a baseline. The baseline consists of tests created as a result of sprint requirements that will be fulfilled.

- **Smoke Testing:** For every sprint, we review the new test cases and adjust standard smoke tests to reflect any needed changes.

- **Regression Testing:** For every sprint, we review the new test cases, and based on the requirements and development, we complete a regression test. That full regression test includes all the sprint baseline regression and smoke tests. The regression test is fully automated with *QTP*.

- **Load Testing:** During sprints at Herndon, based on the needs, we develop LoadRunner scripts for performance and tuning. At the end of each iteration, our goal is to have a set of LoadRunner scripts that will allow us to see the performance, load and scalabity for major business rules and transactions or identify bottlenecks...

- **Service Testing:** During sprints at Herndon, based on the needs, we develop services tests scripts. Our goal is to run these scripts under load as well as security.

In One Iteration: Smoke and Regression Tests

- Total Number of Test Cases
    - Smoke – 87 per browser
    - Regression – 259 per browser
    - Patch – approx 15 per browser

- Total Number of Releases
    - 18 so far, with additional releases to some rounds

- Days and Resources to Test
    - Smoke - 3.5 hours per browser (uninterrupted), usually 1 person
    - Regression - 18 hours per browser (uninterrupted), usually 2 people
    - Patch - 4 hours per browser (uninterrupted), usually 1 person

- Number of cycles before ATRR
    - Herndon
    - Client Suite A
    - Client Suite B

- Hours of Smoke, Patch and Regression
    - Herndon – 4 days per release
    - Client - Suite A – 3 days per release

- For each release, we have ~1000 test cases to be executed, with 5 to 6 resources over 200 hours each.

- This does not include downtime or any technical problems.

- With the following assumptions:
    - The testers are very familiar with the system.
    - The system is already set up per the requirements and test cases are provided.
    - If a test fails, defects descriptions and defects are already in the system.

Believe it or not testing takes a lot of time and resources!

- **Test Plan**
  - Living document updated throughout iteration
  - User stories augment the test plan
  - Delivered at the end of each iteration

- **Test Cases**
  - Written throughout the Agile process
  - Input to Rational prior to end of each sprint
  - Automate QTP, LR and ST

- **Test Results**
  - At Herndon with Agile Teams
  - Delivered at the end of each sprint

- **System Test Report**
  - Living document updated throughout iteration
  - Updated at the end of each sprint
  - Delivered at the end of each iteration

Script Development

- Developed global scripts that can be called from one script to the other
- Scripts were grouped into test sets for different purposes. Such as quick regression test sets, quick check for critical areas or known issues
- With one script we were able to test the system with different browsers at the same time. E.g., IE6, IE&, FF3, etc.
- The same scripts were used for executing tests at different suits. So with one script we were able to run several tests depending on the situation we were in for that particular day.

Calling scripts from other test sets, excluding log in and log out, preparing test sets and making sure users have the right privileges to perform certain business rules

17

Quality Center, Schedule QTP Scripts, Defects

Questions a Performance Test Should Answer:

- Does the application respond quickly enough for the intended users?

- Will the application handle the expected user load and beyond?

- Will the application handle the number of transactions required by the business?

- Is the application stable under expected and unexpected user loads?

- Are you sure that users will have a positive experience on go-live day?

Performance & Load Test Process in the Application and at Northrop - Herndon

While running performance test scripts one can see the actual response time and monitor the servers involved in the architecture.

21

- **SiteScope Monitors:**

Challenges of Testing for Agile and SOA

- **A more versatile test-bed environments**
  - It may be difficult to model the whole set of end-to-end software that probably span many different servers
  - Ability to simulate unavailable components

- **Transition for testers – process-centric testing team**
  - Broad knowledge of business processes
  - Understanding the intricacies of domino effects on business transactions
  - Cross-functional teams environment
  - Understand and diagnose underlying technology and connectivity

- **Location and identification of web services (Geographic locations…)**

- **Availability of web services components: Applications, Middleware, Supporting hardware, teams – development, system admin, network, etc.**

- **Locating and isolating defects are difficult:**
  - **Defects in service components would cause domino effects to applications that utilize those services**
  - **Capture and analysis of all SOAP messages that are passed from one component to another is overwhelming**
  - **Service components do not have GUI**

Positive Testing - Generates a full positive test for the selected services. It tests each operation of the selected service.

Standard Compliance – Tests the service's compliance with industry standards such as WS-I and SOAP.

Service Interopera... ...with all supported Web...

- .NET Frame... ...ations using .NET Frame...
- Axis 1.3 We... ...ations using Axis 1.3 We...
- Generic Mer... ...sing Generic Mercury Sol...

Security Testing –

- SQL Injectio... ...es it an unauthoriz...
- Cross... ...site that will distu...

Boundary Te... ...nique.

- Extreme Va...
- Null Values

WSDL and URL based service

24

Map Service Interface Description

The application has three Mapping Interfaces:

- Map Server

- Reporting Detail Request from Palanterra

- Reporting Mapping Call

Mapping Thrid Party

Application

| Map Server Request |
| Map Server Response |
| HTTP Request |
| HTTP Response |

Reporting Mapping Call

NORTHROP GRUMMAN

Report Mapping Call is the single internal Mapping interface in the application.

When a user clicks the "Map" button within the results page of a report, a call is made to the Mapping application.

Mapping	User	Reports	Mapping related servlets	ESB	Application

GetCapabilties

CapabiltiesInfo

GetMap

MapInfo

Available Graphs ✕

- Runtime Graphs
  - Running Vusers
  - User Defined Data Points
  - Error Statistics
  - Vusers with Errors
- Transaction Graphs
  - Trans Response Time
  - Trans/Sec (Passed)
  - Trans/Sec (Failed,Stopped)
  - Total Trans/Sec (Passed)
- Web Resource Graphs
  - Hits per Second
  - Throughput
  - HTTP Responses per Second
  - Pages Downloaded per Second
  - Retries per Second
  - Connections
  - Connections per Second
  - SSL per Second
- System Resource Graphs
  - Windows Resources
  - UNIX Resources
  - Server Resources
  - SNMP
  - Antara FlameThrower
  - SiteScope
- Network Graphs
  - Network Delay Time
- Firewalls
  - CheckPoint FireWall-1

Apache - Last 60 sec

UNIX Resources - Last 60 sec

Trans Response Time - whole scenario

Hits per Second - whole scenario

Color	Scale	Transaction	Max	Min	Avg	Std	Last
	1	01_MCAT CBT Home	3.929	0.268	0.742	1.076	0.303
	1	vuser_init_Transaction	0.001	0.001	0.001	0.000	0.001
	1	02_Login	0.635	0.215	0.318	0.106	0.231
	1	03_Begin Registration	0.668	0.154	0.241	0.122	0.160
	1	04_Submit Bio Info	1.040	0.149	0.265	0.210	0.194
	1	05_Search School by State	0.447	0.276	0.332	0.061	0.327

Design   Run

With Service Test We are Able to:

- **Develop scripts:**
  - without a GUI
  - using multiple protocols. In enterprise world we have to deal with a lot of multiple protocols. This feature is very helpful.
  - by WSDL, UDDI, File and URL. This is a very helpful feature, too.

- **These scripts can be executed in LoadRunner for performance**

- **We can analyze traffic over the network**

- **We can set Security Policies that includes tokens, SAML, and so on**

Start early in the life cycle:

- Testing client applications – Start the end-to-end testing and tuning 6 months before the deployment.

Create an assembly-oriented test plan

- Test the permutations incrementally and combination of paths – Divide testing into stages with incremental increase the number of components.

- Choose an appropriate set of test cases to support end-to-end testing of the business process and end-user experience.

**Create an automated and repeatable testing process.
High number of permutations and combination of paths!**

With the use of the tools we were able to:

- Prioritize testing priorities based on business risk

- Access testing assets anytime, anywhere via a browser interface

- Create an end-to-end quality management infrastructure

- Manage manual and automated tests.

- Accelerate testing cycles by scheduling and running tests automatically, unattended, 24x7

- Manage multiple versions of requirements, tests, test scripts and business components

- Enforce standardized processes and best practices

- Analyze application readiness at any point in the testing process with integrated graphs and reports

With the use of the tools we were able to:

- 50 to 70% decrease in actual testing time (efficient and faster)

- Able to cope with huge amount of testing and captured defects at early stages of development

- Able to produce contractual documents such as RTM – Requirement Traceability Matrix, defect reports, test reports, test plans and the like in a timely manner.

- Able to produce metrics for defects such as defect shipment density, defect aging and other metrics related to defects

- Were able to capture change before third party test teams and sub contractors in Serv Te

- Able to c ate ie

- Able to ca ec

- Had a ch un of performance bottlenecks in architect ab o nce of the system

- Most imp d ed to have the same set up that we have at Herndon ead of sending testers to classified lab we are planning to send sc is the plan…

What We Need For Contractor Integration Results

Purpose: integrate and test all system components prior to official delivery to the government.

- ✓ All testing related documentation has been completed and is up to date
- ✓ Successful completion of smoke, patch, and regression tests
- ✓ Performance-Load-Stress test baselines obtained
- ✓ SLAs are met
- ✓ All the test results were delivered into government CM
- ✓ All defects are documented in CM tool
- ✓ Final system test report submitted to the PMO
- ✓ Installation and build guide with all the updates completed and delivered

Results from test indicated that the software is ready for Acceptance test

Requirements Validation/Regression Testing Results

NORTHROP GRUMMAN

Requirements Validation

Results we would like to see:

- Release Requirements testing
  - Completed 99.13% testing of all testable requirements
    - ✓ Executed 99.63% testing against IE 6.0
    - ✓ Executed 98.50% testing against IE 7.0
    - ✓ Remaining 0.93% of testing could not be functionally tested

- Regression testing
  - Completed 100% of planned regression testing
    - ✓ Executed 95.05% testing against IE 6.0
    - ✓ Executed 92.95% testing against IE 7.0