API Testing – Picking The Right Strategy

Asha KR
Shwetha D J
Bio

Asha KR, Senior Software QA Engineer at Intel Security, currently working in the Intel Security India Center in Bangalore. Working for the past 5+ years in different QA roles in SaaS products. Asha holds Bachelor of Engineering in E&C from VTU, Karnataka, India.

Shwetha DJ, Software Development Engineer for Test, currently working in the Intel Security India Center in Bangalore. She has been working for the past 1+ years on SaaS products. Shwetha holds Bachelor of Engineering in E&C from VTU, Karnataka, India.
Introduction
The Problem
The Solution
The Approach
Results
Conclusion..
- About Web services
- About Application Programming Interface (API’s)
- What is API testing
- API testing in project architecture
- Steps involved in API testing
STEPS
API TESTING

PNSQC 2015
Crucial integration Testing

UI failure

Time constraint in releases

PROBLEMS

Manual effort maintaining product
The Solution

- API Testing - Manual
  - WCF Test Client
  - Soap UI

- API Testing - Automation
  - Soap UI
  - Microsoft Web Test
  - Using Programming Language
WCF Client

- Windows Communication Foundation
- Microsoft .NET framework and user friendly
- Give input, invoke method and analyze results

Advantages

- Interoperability
- Security and Reliability

Disadvantages

- Pending requests aborted when closed
- Automatically generated values and cumbersome to use
WCF Client

Authenticate

Request

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>request</td>
<td>AuthenticateRequest</td>
<td>AuthenticateRequest</td>
</tr>
<tr>
<td>RequestId</td>
<td>cc98885-c752-430e-831a-f1c3db2e7754</td>
<td>System.Guid</td>
</tr>
<tr>
<td>CallerApplicationName</td>
<td>(null)</td>
<td>System.String</td>
</tr>
<tr>
<td>CallerApplicationVersion</td>
<td>(null)</td>
<td>System.String</td>
</tr>
<tr>
<td>CallerHostName</td>
<td>(null)</td>
<td>System.String</td>
</tr>
<tr>
<td>CallerId</td>
<td>(null)</td>
<td>System.Nullable&lt;System.Guid&gt;</td>
</tr>
<tr>
<td>SessionId</td>
<td>123456789</td>
<td>System.String</td>
</tr>
<tr>
<td>AccessKey</td>
<td>(null)</td>
<td>System.String</td>
</tr>
<tr>
<td>LogonName</td>
<td>SupportAdmin</td>
<td>System.String</td>
</tr>
<tr>
<td>Password</td>
<td>welcome</td>
<td>System.String</td>
</tr>
<tr>
<td>UserLocale</td>
<td>(null)</td>
<td>System.String</td>
</tr>
<tr>
<td>AllowOtpAuthentication</td>
<td>False</td>
<td>System.Boolean</td>
</tr>
</tbody>
</table>

Response

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>(return)</td>
<td>(null)</td>
<td>AuthenticateResponse</td>
</tr>
<tr>
<td>ResultCode</td>
<td>0</td>
<td>System.Int32</td>
</tr>
<tr>
<td>ResultMessage</td>
<td>&quot;Success&quot;</td>
<td>System.String</td>
</tr>
</tbody>
</table>
The Solution – Soap UI

- Simple Object Access Protocol
- Communicates over HTTP
- Suite execution using user interface
- Suite triggers from automation build / command prompt

Advantages
- Extend HTTP for XML messaging
- Platform- and language-independent

Disadvantages
- No mention of security facilities.
- Do not hold any state references to remote objects
The Solution – Soap UI
The Solution – Microsoft Web Test

- User performs a set of operations
- Good for simpler web services and straightforward functionality
- Suitable for performance testing

Advantages
- Easy test scripts creation using test recorder
- Edit recorded tests to tailor to specific needs

Disadvantages
- Not much validation logic can be put into the tests
- High Volume tests were not at all executed in the test suite
- Maintenance is difficult
The Solution – Microsoft Web Test
Current Approach – Testing Framework

- Stable automation framework using **C#**
- Simple, object-oriented language
- Test Framework
Current Approach – Testing Framework

Advantages

- C# framework is tool independent
- Ease-of-development, rich class library, functions
- End to End solution with DB validations
- Anyone can contribute to the framework
- Interoperability

Disadvantages

- Depends on .NET framework. Library not found in .NET difficult to implement
- C# doesn’t come up with open source technologies and operating systems
Challenges in API Testing

- Parameter Combination, Call Sequencing
- No GUI - difficult to provide inputs
- Not easy to validate output
- Exception handling function
Every QA had opportunity to become a Programmer

Stable framework delivered, Quality product to Customer

Deeper insight into the product through API Testing

Percentage of early defect detection: 20%
Automation Test Results

Smoke Tests

- Automated: 100%
- Not Automated: 0%

Total TestCases: 390
  - Automated: 390
  - Not Automated: 0

100% automation of smoke test cases

Functional Test Cases

- Automated: 45%
- Not Automated: 55%

Total TestCases: 1550
  - Automated: 850
  - Not Automated: 700

60% automation of functional test cases

Automated Test Coverage

- BVT: 100%
- FVT: 60%
- End to End Test Cases: 0%

Test Coverage
API Testing -- Advantages

- Much better quality product, ensuring the data access
- Simple security compliance testing
- Complete functionality allows for easy future expansion of the application
- Structured way of testing, makes automation more feasible

API Testing -- Disadvantages

- No Encryption
- Increasing Tooling needs
- Straightforward reason for failing API is not known like GUI
End to End Testing:

- Extension of all BVT scenarios to test end to end functionality
- Service call for different APIs in a single method
- Needs a good coding style and capturing of exact results

Integration Testing

- Testing of different systems as one
- Covering BVT and FVT functionality by considering all system behavior
Thank you!