Whitepaper

Functional API Testing

October 2017

Interra Information Technologies

www.interrait.com

INTERRA Information Technologies

Table of Contents

1. Introduction

The main goal of functional API testing is to validate the functionality of the business logic layer which occurs prior to GUI integration test. Recognizing the common challenges and formulating the best test approach ensures the full test coverage and eventually reduces the number of defects rates throughout the test life cycle. Here we have described the totality of API testing in terms of functionality test throughout width & breadth of it, which we cover and offer to our clients.

2. How important is API Testing?

The importance of functional API testing is described as following with reasons:

- API testing can be done independent of GUI ensuring that business logic is defect free.
- API testing can help detect application behavior problems even before GUI integration & GUI developed.
- Less time consuming than GUI functional testing.
- Lightweight XML or JSON data transfer modes.

3. Challenges that we faced on Testing of API-based Applications

There are certain challenges that we faced during initiation and while performing API testing's mainly from functionality testing perspective. They are as follows.

- No GUI and thus not a fully black box type testing is applicable. Things here to consider additionally always like authorization & authentications, all the parameters and their validations, responses and its time etc.
- When API is the final product, in these APIs expose functionality that the developer hopes or thinks other developers/ users will want. In this case, the tester's role needs also to include a user advocate role, to make sure the offering is complete – everything that any user may need is all there. Functional, security & load testing along with service level documentation to be checked thoroughly.
- The data formatting that handles requests and responses for the API–needs to be maintained throughout the testing process. Any updates to the program that create additional parameters for the API calls need to be reflected in the schema configuration.
- API calls need to appear in a specific order to work correctly. This creates a sequencing challenge for the testing team. If a call to return a user's profile information goes through before the profile is created, the request will return an error.
- Testing teams find validating the parameters sent through API requests challenging as well. The sheer number of parameters and use cases for those parameters can make it a daunting task. The team needs to make sure all parameter data uses the correct string or numerical data type, fits within length restrictions, fits within a designated value range, and passes other validation criteria.

- Increased end-to-end testing for regression, managing smoke testing for new builds , maintenance phase to accommodate frequent changes and releases.
- Verification of business requirements through API.
- Identification of right tools to facilitate testing & choosing suitable test automation framework for functional testing.
- Integrating with Service design document or API documentation for maintenance and schema update.
- To generate a full coverage report and execution summary every time a set of tests is run.

4. Our Approach

In order to overcome the above challenges we organized and created an approach for API based testing which has proved beneficial for the overall business in terms of performing this API test activity. The similar approach is used in one of our client (A giant top ranked Japanese Telecom provider)

- Framework driven API test .
- Layered architecture which takes care of basic Security of each API .
- Parsing mechanism.
- Validations support for each and every parameters .
- Structured test data.
- Reusable test scripts and framework .
- Creation of Individual API as well use case based test suites.
- Smoke , Regression test suites maintenances on each build in parallel .
- Integration with swagger API – changes in documentation is mapped to API test framework.
- Generation of reports and execution log with full request and responses.

Type of tests performed

- Functional
  - Individual API level (Service level test).
  - Use case based scenarios .

- Non - Functional
  - Security .
  - Individual API performance level judged .

Security verification

- Authentication: Identifying the end user with the key .
- Authorization: Providing identified user access to correct resources/data through an access token using private & public key.
- Encryption: Hiding information from unauthorized access.
- Signatures: Ensuring information integrity, so as to check that API requests or response have not been tampered within transit. Short lived and expires automatically after few seconds or accessing it.

Level of tests

- Unit – focusing on each API level with positive & negative cases.
- Integration – focusing on API interactions with different API's of different modules and ensuring smooth transactions of data .

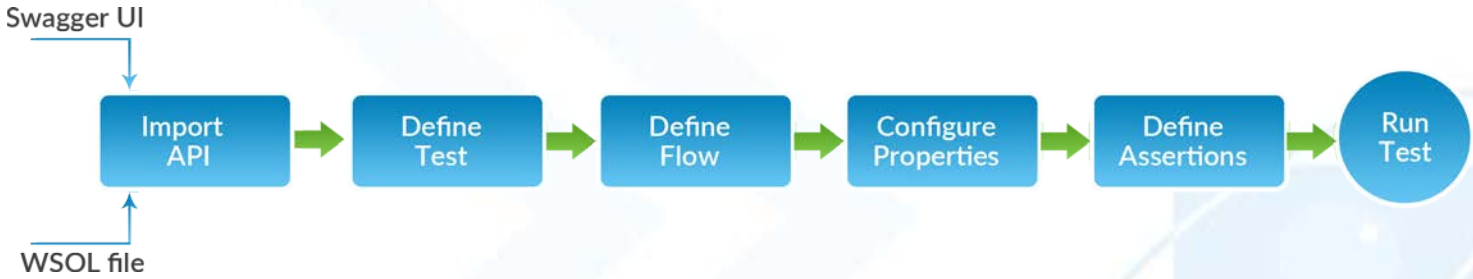
5. API test tools, writing tests & test framework used

Test tool used: SOAP UI

In SOAP UI there is a free version where reporting and execution history is not there.

We created our own test scripts and framework to generate those as well and utilized the tool to its maximum without using licensing cost in it.

Stages of writing a test for an API. Once an API is complete this can be taken into different suites based on requirements (like smoke, regression etc.).



Framework

The SoapUI Automation Framework provides

- Easy-to configure, reusable SoapUI project file.

- **Groovy**: It includes Groovy script in the base project that generates signature for authentication & successfully executes the API .It also reads & write data to and from Excel file/ txt file and executes the test cases using the SoapUI tool.

- **Assertions**: The test cases validate the functionality of the Web service or services by sending the XML-formatted requests as specified in the test case and verify (through proper checkpoints called assertions) the response accordingly with various types of assertions as per nature of the request parameter & response.

- **Properties**: It allow users to configure and use specific value in different test steps within Test Suite. To specify the configuration values used within the Groovy test scripts, users can create properties in Test Suite and at the test-case level with a base project. The values can be referenced using either property transfer or property expansion mechanisms.

- The Automation Framework has pre-defined properties that help the Groovy scripts to set configurations with the values test engineers created in the Test Suite.

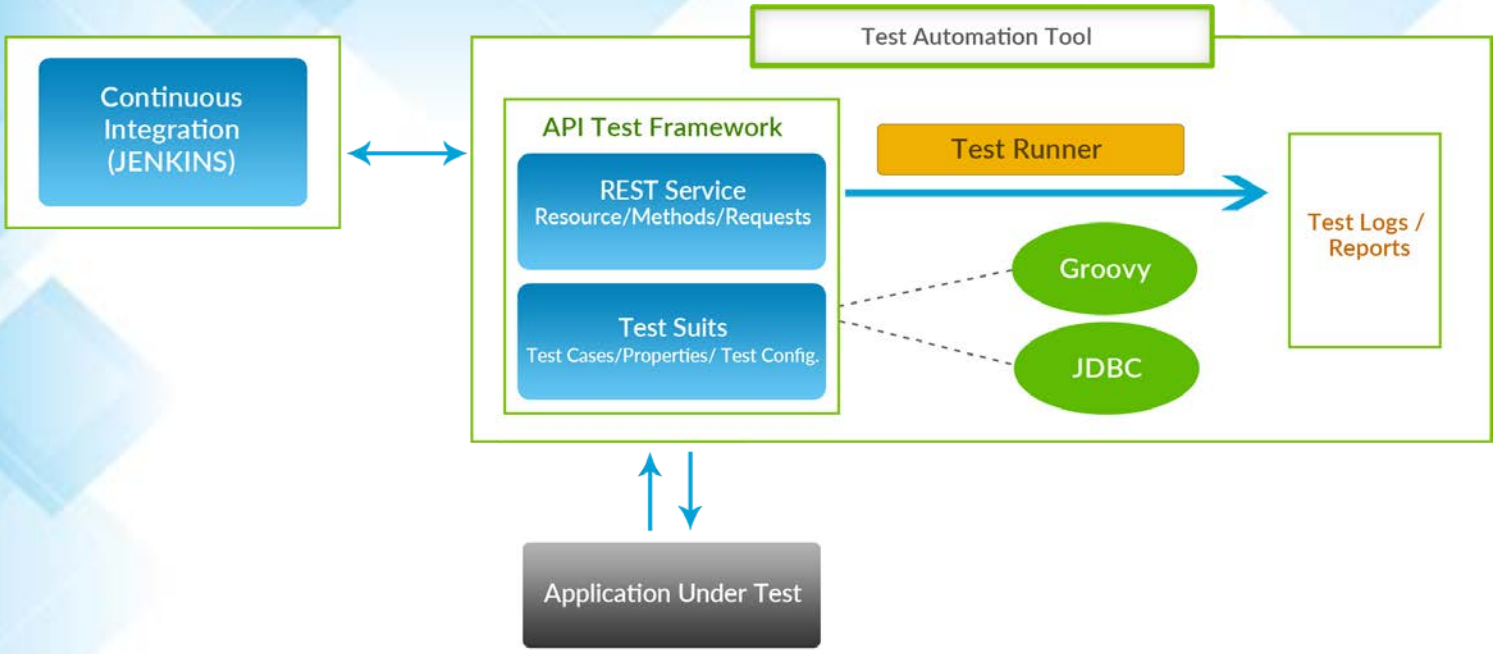
- **JDBC** : Using JDBC the data in the response is also checked and validated through the database.

- **Test runner :** The test runner allows to run SoapUI tests and export results from command line as well .

- **Reports/Logs:** For every test run the logs is generated and stored in a specified location as mention in the configuration. The log contain all the request and response on each run which stored with time stamp. There is an excel report at the end confirm the coverage of the total API executed and its overall pass/fail/not run status .

Benefits

- Provides a fast solution for easy development and automated execution.
- Reduces human efforts.



Techniques

- Groovy script for data transfer.
- Running test suite using testrunner.bat.
- Storing the results dynamically using timestamp.
- Data checking through JDBC.
- Generating execution log and test summary report through custom code (Groovy script).

6. Benefits achieved

Following are the benefits that we achieved while applying our test approach in large project for Japanese telecom giant client.

- The business logic got verified timely before any GUI tests & any pending GUI activities .
- Reduced the manual effort of smoke/regression testing to considerable amount for each sprint/build and to judge the quality within short time.
- There are around 350 API's for which 1000+ tests are written and executed within few hrs. through SOAP UI test framework .
- Reduced licensing costs by using open source tools for functional and load testing.
- Critical business use case verification from SOA end easily and quickly.
- Easy test data creation for debugging , test execution and support related activities.

7. Best Practices

- **API Documentation** – Emphasis on service design document so as to know in detail what, how and when an API is executed. Doing so makes readability, as well as maintenance, far easier tasks. Periodic review and updates are a part of test cycle .Moreover this gives a better test scope and approach .

- **Extent of Automating the API's** – Avoid testing cases that are likely to cause serious problems, like crashing for instance. Although tackling these first is fairly typical, it isn't the best way to prioritize testing efforts. Rather, begin with mainline business critical cases / repeated case which impacts directly in the production roll out is to be considered for a test suite.

- To adopt the following **guidelines** so as to meet the criteria of fast and quality testing like
  - ❖ Creating as much as possible global variables at suite level for easy debugging & change.
  - ❖ Creating generic test cases which can be re-usable and can save time for creating different test suites.
  - ❖ Emphasizing on naming convention so as to better understand and readability.

- To adopt the **type of test techniques** which fits the overall business requirement in terms of API like
  - ❖ Syntax testing of individual operations or methods.
  - ❖ Functional testing on individual operations or methods.
  - ❖ Scenarios based testing focused on business operations once above two is completed.

- Uses of Proper **Assertions** - Try to make an assertion that covers the whole message, this makes the assertion complete and reliable.

- Use of **Groovy script** – make a general scripts for authentication and authorization and try to reuse it which will in turn reduce the number of test steps .

- Use **modular** and **scalable** approach while designing the test framework .

Credits
Web Services Automation Testing using SoapUI Tool - > International Journal of Innovative Engineering Research

International Journal of Innovative Engineering Research -> QASource

http://www.ttii.co.kr/solution/pdf/Getting_Started_with_API_testing.pdf

ABOUT INTERRAIT OFFERINGS

InterraIT is a leader in business consulting and technology solutions. We develop path breaking solutions in Digital Transformation, Cloud, Cognitive Computing, ServiceNow, Big Data Technologies and Mobility to help our clients adapt to the digital world and make them successful.

ABOUT INTERRAIT

InterraIT is a global IT consulting & services company providing business software solutions that help clients drive innovation, improve operational efficiencies and address organization's unique business needs at lower TCO & faster ROI. Established in 1996 & headquartered in San Jose, InterraIT has world-class delivery centers in India & has client-partner network spanning across the United States.

Stay Connected: **Email:** Sales@interrait.com www.interrait.com

Corporate Office
2099 Gateway Place, Suite 200, San Jose, CA 95110
Tel +1 408 451 1700, Fax +1 408 441 7495

