Smart Assistant for Intel System Integration and Validation

Submitted By

Nirali Kalariya

18MCEN05

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

MAY 2020

Smart Assistant for Intel System Integration and Validation

Major Project

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering

Submitted By

Nirali Kalariya

(18MCEN05)

Guided By

Prof. Jigna Patel

Nirma University, Ahmedabad.

Ms. Manjula K M

Intel Technology India Pvt. Ltd.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

MAY 2020

Certificate

This is to certify that the major project entitled **"Smart Assistant for Intel System Integration and Validation"** submitted by **Nirali Kalariya (18MCEN05)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering(Networking Technology) of Institute of Technology, Nirma University, Ahmedabad, is the record of work carried out by her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Prof. Jigna Patel
Internal Guide & Assistant Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Prof. Gaurang Raval
Associate Professor,
Coordinator M.Tech - CSE(NT)
Institute of Technology,
Nirma University, Ahmedabad

Dr. Madhuri Bhavsar
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.

Dr. R. N. Patel
I/C Director,
Institute of Technology,
Nirma University, Ahmedabad

Statement of Originality

I, **Nirali Kalariya**, **18MCEN05**, give undertaking that the Major Project entitled "**Smart Assistant for Intel System Integration and Validation**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

_____

Signature of Student

Date:

Place:

Endorsed by

Prof. Jigna Patel

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Prof. Jigna Patel**, Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for hervaluable guidance and continual encouragement throughout this work. The appreciation and continual support she has imparted has been a great motivation to me in reaching a higher goal. Her guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

It gives me an immense pleasure to thank **Dr. Madhuri Bhavsar**, Hon'ble Head of Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for her kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr. Alka Mahajan**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

- Nirali Kalariya

18MCEN05

Abstract

PC platform validation requires to choose set of optimal test cases and prioritize each of the test case and then executes those test cases on weekly basis to suitable the operating system and software requirements. More than three thousand test case need to be run on a weekly basis, so filtering the set of optimal test cases among all which are best suited for run is difficult task. It is important to find a simple, standardized and consistent approach that can be applied across multiple products in order to improve efficiency and consistently scale across products. We suggest a simplistic approach to help identify suitable, cost-optimized test cases to run in a continuous environment without sacrificing validation coverage or identification of defects.

Abbreviations

SSMS	SQL Server Management Studio
ETL	Extract, Transform and Load
CSV	Comma Seprated Values
PDF	Portable Document Format
XML	Extensible Markup Language
HTTP	Hypertext transport protocol
SQL	Structured query language
HSD	High Speed Database
UX	User Experience
DB	Database
OS	Operating System
SSD	Solid state drive
HDD	Hard disk drive

Contents

List of Figures

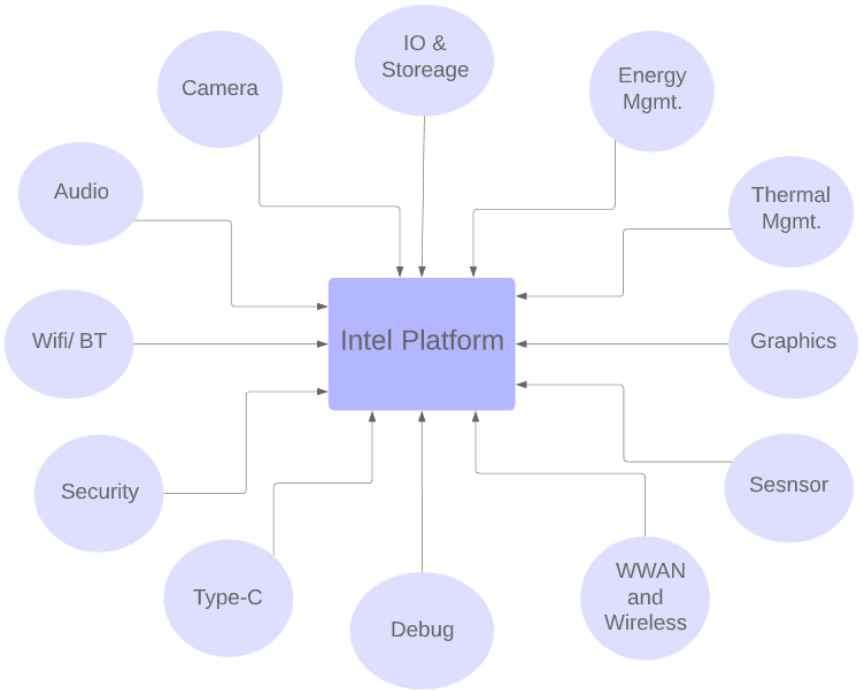
Chapter 1

Introduction

1.1 Overview

The PC architecture is a mix of components of hardware, software and firmware that provides compelling functionality and end-user application models. The components of the platform are BIOS, chip firmware and system drivers that allow multiple components such as Audio, Graphics, Display, Wi-Fi, Bluetooth, Camera, Wireless WAN, HSD, SSD, Sensors, Protection, etc.

Figure 1.1: System Components



The Client Computing Group(CCG) in Intel is responsible to deliver all aspects of the client computing business like laptops, Smart Phone, Tablet and PC platforms. With goal of providing high quality Intel platforms to the clients. To achieve functional requirements and development we need to do system integration and validation.[1]

On average, a framework includes over more than 40 component elements from the platform, more than 1000 functional specifications from the platform and more than 2000 test cases from the system that need to be performed to verify the overall functionality.

Mainly focus on Platform integration and run different-different test cases to validate new changes on the Intel platform. We run test cases to detect defects of Intel platform when all components integrated to one platform. So, All components can be integrated in time.

Chapter 2

Literature Survey

2.1 Motivation

Time to market is key requirement for Intel CCG team. Client Computing Group is primarily known as Intel's PC and Laptop business. To achieve time to market and performance of Intel Processor Products we need to drive innovation across the Platform. Mainly in System Integration and validation, we need to come with proper automation using which we should able to validate entire platform with 100% coverage of testing with minimum time cycle. So, we can avoid all manual work done by System Validation Engineers and reduce delivery time of the platforms. We need to rethink entire work process to achieve this goal. Using this automation can help to reduce human errors, remove unwanted test runs which already executed in previous release cycle. We need to come up with smart algorithms which can take own decision for no of test cases and generate configuration automatically. We should have advance dash board, using it Program/ Project manager or higher authority executives can take organization decisions easily for System integration and validation of platforms.

2.2 Significance

The system validation is designed to help any original equipment manufacturer (OEM) / Original design manufacture (ODM) delivers compatible and reliable systems, hardware, and software products. Compatibility and reliability are assured by end-users as a result of their trust for the logo. WHCK (Windows Hardware Certification Kit) is intended to help develop systems and devices that have been tested to ensure that they meet

Microsoft standards for Windows 10 onwards as well as the quality level that ensures a great Windows experience for end-users.

2.3 Limitations

- Intel Smart Assistant is targeted for Windows Operating system only in Intel Platforms. We can't use it for other OS like Linux, Android.
- An Application can generate the wrong configuration if Lead's inputs are given wrong.

2.4 Delimitation

- Continuous feedback from end user will be taken as it follows scrum process model.

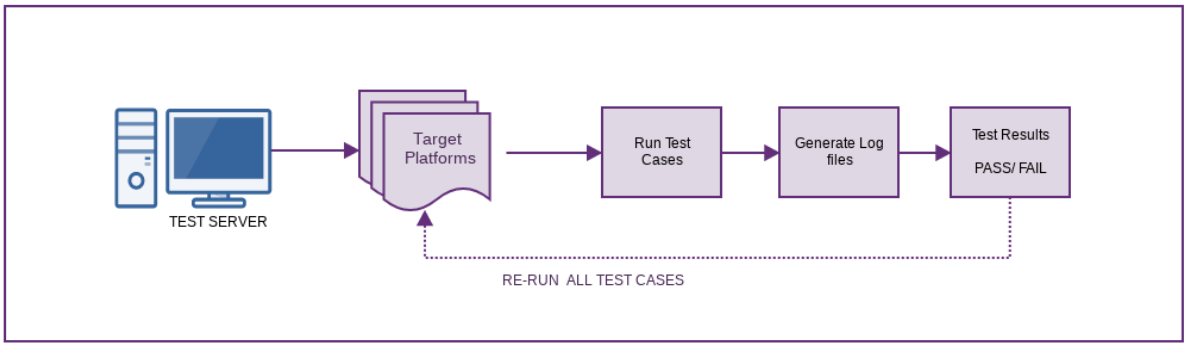
Chapter 3

Existing System Integration and Validation

3.1 Manual Work

Test server will performs a series of test cases and produce individual log files which needs to be inspected manually to pinpoint errors and debug for the subsequent root cause of the failures. The tool kit does not have inherent intelligence to identify the cause of failures. This is a tedious process and at the same time depends upon the individual debug skills.

Figure 3.1: Manual flow of running Test cases



Using manual work for tracking activities, bugs/issues, validation, verification, etc. Manual work is good for small/less complex projects but Intel's business processes are very complex and manual work way of maintaining is labor intensive and error prone.

Below are the main limitations in the above manual process:

- **High Time to market** - Required too much time to do system integration and validation for any new next generation of Intel platforms. Really complex and difficult to integrate all system components.
- **Redundancy** : Running same test cases more than once which is not required every week.
- **High Resource allocation**: Back and forth follow up with Software/ Hardware/ Firmware engineers. Difficult to get closures on time.
- **High Complexity**: Need to debug each and every test cases manually. No systematic way to debug using any automation.
- **No scalability**: Dependency on engineers for multiple system integration & validation at the same time.
- **No Dashboard**: Difficult to track each release live status from validation engineers.
- **Less efficient** due to human errors.
- **High Dependency** on all engineers.

Chapter 4

Proposed work

4.1 Introduction

The intention of continuous chip integration and validation is to incrementally enable features and produce a quality weekly software stack that can be consumed through various production version milestones by internal and external customers. To validate the application stack, as part of the continuous integration and verification process, the system validation team performs approximately more than 3000 test cases on a weekly basis. The secure execution technique is to run all the tests on a weekly basis to ensure full coverage, but it is sub-optimal, considering that there are some test cases that would never have failed in run history and it is costly to run them over and over. Based on this, the challenge is to find the best way to run optimal test cases on a weekly basis without sacrificing on the ability to detect defects. [2].

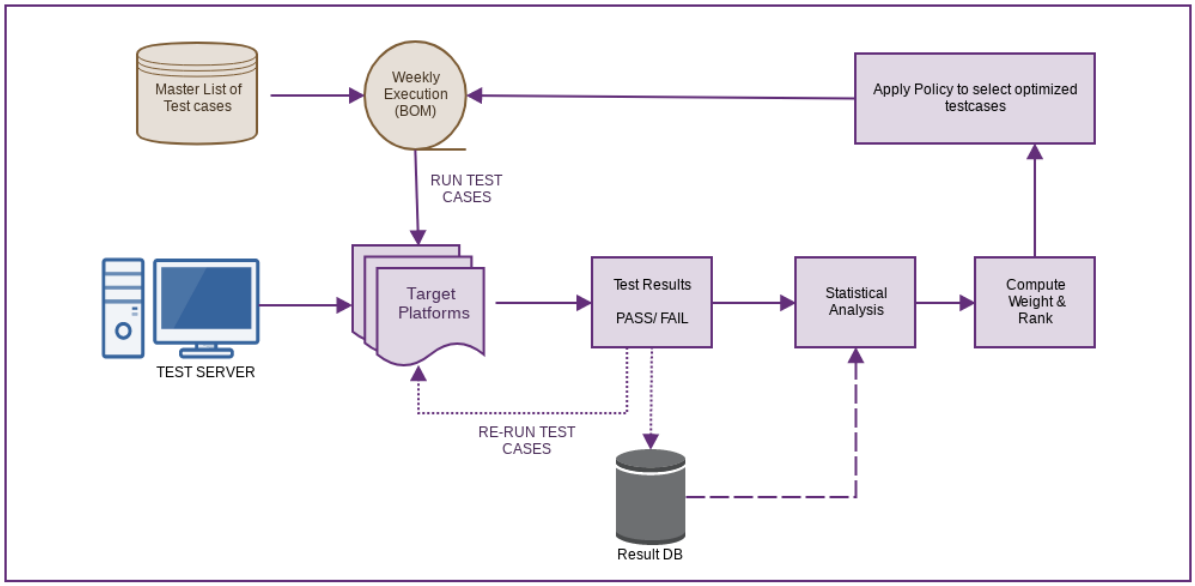
4.2 Proposed Solution

There are a number of techniques that are investigated to suggest in the past to extract optimum collection of test cases and the two common techniques that are prominent. First is a based on software change testing and second is a prioritization based on historical pass or fail data. [3]

The source code update based software testing is impractical from a system validation perspective, given the volume of changes in source code across 40 plus components

without adequate instrumentation. The system software also primarily consists of device drivers and firmware, and these components form the fundamental basis for running operating system and applications, and thus any driver changes affect basic flows.

Figure 4.1: Automated flow of running Test cases



The solution is to feed the weekly findings into a process of statistical analysis to classify the test cases that have a major impact in finding flaws. The approach also allows an acceptable strategy of optimization to be chosen depending on the system process.

The plan helps to determine the number of test cases to be carried out. Over optimization will lead to faulty escape and, conversely, the cost of execution will be burdened. This reinforcement-based learning cycle is repeated throughout the weekly test execution until the duration of the project ensures consistent optimization methodology.

Therefore, based on this assumption and from a project viewpoint, the history-based fail / pass prioritization of test cases tends to be an appropriate approach for determining the best set of test cases to run over platform peaks.[4]

So, The solution is based on the following key characteristics

- Test case's results (pass, fail, not run, blocked).

- Failure ratio.
- The solution also allows to select an appropriate policy and identify the test cases that need to be run based on failed components for particular platform.

Chapter 5

Technical Requirements

5.1 Technical Requirements

- **Windows 2010 R2 64 Bit Server OS** - We have used Windows Server OS for Test Server. Each Test server will connect to targeted machines and will execute the test cases.

- **Intel Atom & Core Processor** - Targeted machine will have next generation Intel Atom and core i3, i5, i7, i9 processors. This is main machines which required to tested.

- **Visual Studio 2019 - ASP.NET C#** - We have used Visual Studio IDE to create web based application for our smart assistant. It will have different pages like - Add project, Add SKUs, Add BOMs, etc.

- **Microsoft SQL Server Management Studio** - We have used SQL server as database for our smart assistant. We have used Management studio tool as GUI client to create databases, tables and store procedures four smart assistant application.

- **Python 3.6** - Each component analysis is done in Python language using Anaconda packages(pandas, scipy).

- **Jupyter Notebook** - Using Jupyter notebook, we are able to create web application for analyzing the each components like USB, BIOS, Sensor, Wifi, BT, Audio, etc. So, we can easily create and share python notebooks with team members.

- **Git** - Using Git we are maintaining the code version of our web development.
- **Gitlab** - We use Gitlab as server for code collaboration. Using gitlab we can easily tag new releases for the tool. We can pull request changes from other developers.

Chapter 6

Data requirements and sources identified

6.1 Data requirements and sources identified

Each and every business units are using different types of data-sources, based on the requirements we are using real-time raw data from,

- Intel's relational databases
- Internally/third-party developed tools
- Excel files created by employees
- XML files generated from automated tools
- CSV files generated from online automated query

Chapter 7

Analysis of components

7.1 Algorithm for computing the correlation

Linear regression can be used only to calculate the correlation between two or more continuous features of the Dataset. This is only suitable for those features that have an almost linear relationship. But to find a correlation between binary and continuous data we should need to use "point bi-serial correlation" because using just linear regression it's not possible to evaluate the correlation between one continuous feature and the other is binary feature values.

7.2 Point Bi-serial Correlation

The point bi-serial relationship coefficient is the same as the Pearson relationship coefficient utilized in the linear regression technique (measured from -1 to 1). The as it were distinction is we are comparing dichotomous information to continuous information rather than continuous information to persistent information. From this point on let's expect that our dichotomous information is composed of things from two groups of binary values like group-0 for all 0 values in field and group-1 for all the record with having 1 field value and the persistent information as "y". The coefficient is computed by the equation underneath:[5]

Figure 7.1: Point Bi-serial Correlation Formula

$$r_{pb} = \frac{M_0 - M_1}{s_y} \sqrt{\frac{n_0}{n} \frac{n_1}{n}}$$

Where:

M0 = the mean of the data from group 0 (no component change) .

M1 = the mean of the data from group 1 (component change will be there).

Sy = the standard deviation of the continuous data(Fail ratio of testcasees).

n0 = the number of items in group 0 (total testcases with 'No' component change)

n1 = the number of items in group 1 (total testcases with 'Yes' component change).

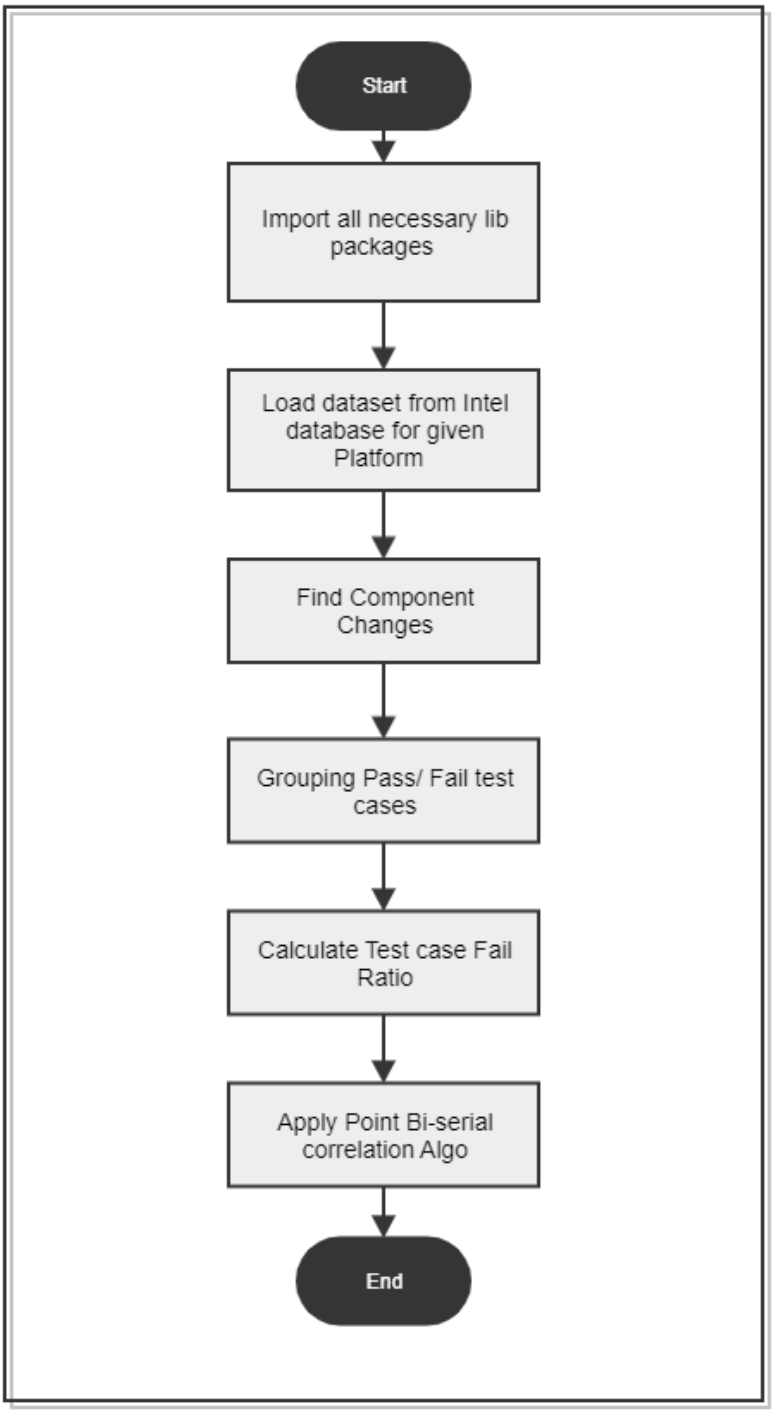
n = the number of items in both groups together (aka the total rows in the data set).

7.3 Components testcase's analysis steps

- Import all the necessary library packages and sklearn machine learning library package for Python into the code of test case analysis.
- Load the dataset into analysis code from the Intel's database which name has been given by the each platform owner.
- Find the component change for each single platform and also across the platform.
- Grouping the data according to pass/ fail test cases of the each platform and count them.
- Calculate the pass/ fail ratio of the each of the test case and according it find number of test cases which are failing for 0% and then calculate the ratio of fail test cases.

- Apply the Point Bi-serial Correlation coefficient between the component change and fail ratio to find correlation between the failing ratio of component test cases and the component change across the platform.

Figure 7.2: Test case analysis flowchart



7.4 Implementation of algorithm in data analysis

From the two data groups pass and fail test cases, segregate the test cases according to component change.

Select two appropriate attribute values to apply the point bi-serial algorithm for finding the correlation between testcases.

Out of them, one is continuous value attribute of the fail ratio of the similar test cases from one platform to another platform or across the platform and the second attribute having dichotomous value attribute of change of component from one platform to another plate form which is having "Yes" or "No" values.

Sample code snippet:

Figure 7.3: Snippet 1

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import re
```

Figure 7.4: Snippet 2

```
# Find rows for particular testcase
df = all_platforms_USB_testcase_ipchange_data.loc[(all_platforms_[redacted]TestCase == testcase)]
row.append(platform_count)
print('{0:20s}: {1:s}'.format("Platforms run on", str(platform_count)))

# Find pointbiserialr for each testcase list
biserialr_result = stats.pointbiserialr(df['[redacted]Change'], df['TestCase_Failed_Ratio'])
row.append(biserialr_result[0])
row.append(biserialr_result[1])
print('{0:20s}: {1:s}\n'.format("Result", str(biserialr_result)))
rows.append(row)
```

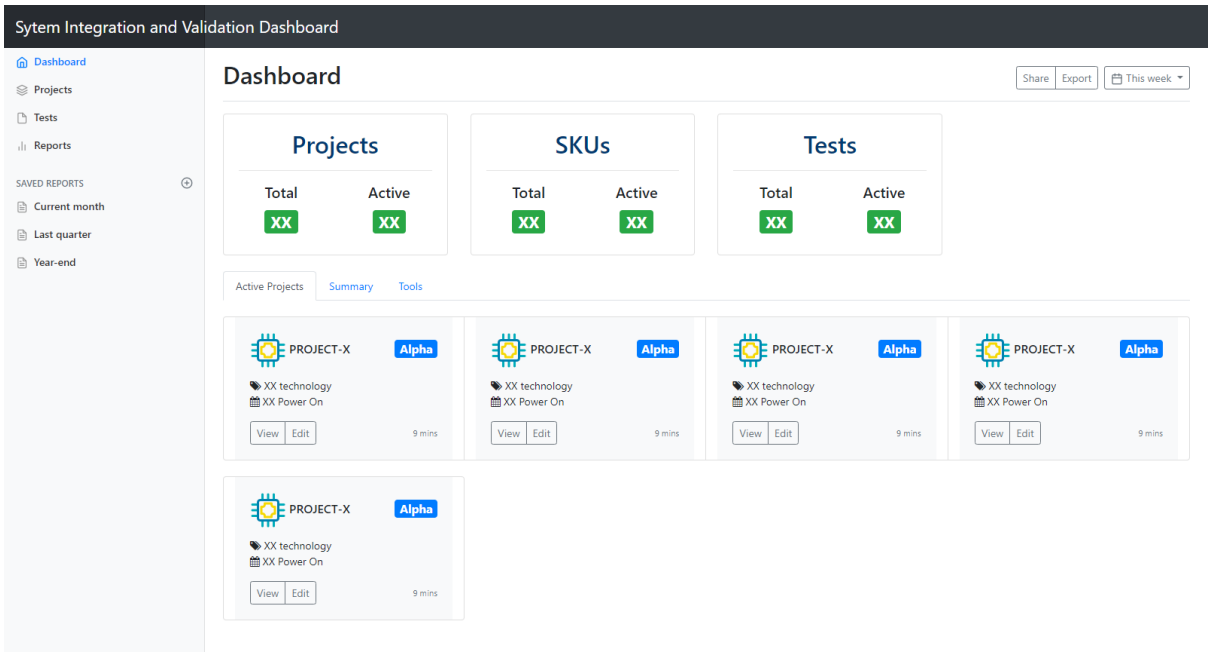
Chapter 8

Implementation

Smart Assistant is Web based application which implemented in Asp.net c# technology stack. Using this web portal any user can go and add new program, SKU.

Using smart assistant we can add any new components. After adding components we can add to the main BKC.

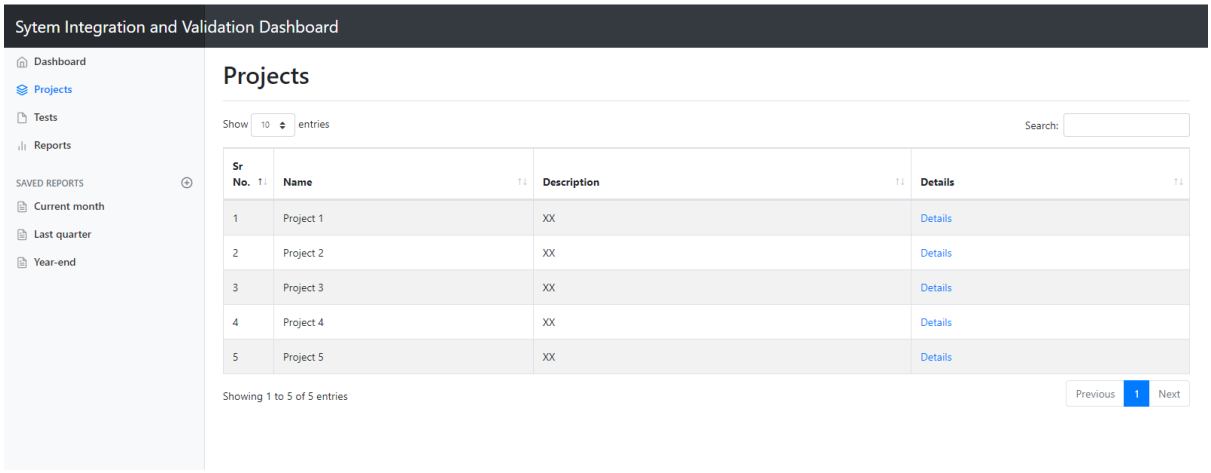
Figure 8.1: Smart Assistant Dashboard



Once the setup is ready, the user performs the following functions.

- Create package where in the test cycle details such as program name, sku name, OS, configuration would be saved.
- Select the targeted machine where he/she wants to run the test.
- Select the type and number of test cases which will run on the above selected target machine.
- Confirm and execute the final set of test cases.

Figure 8.2: Project



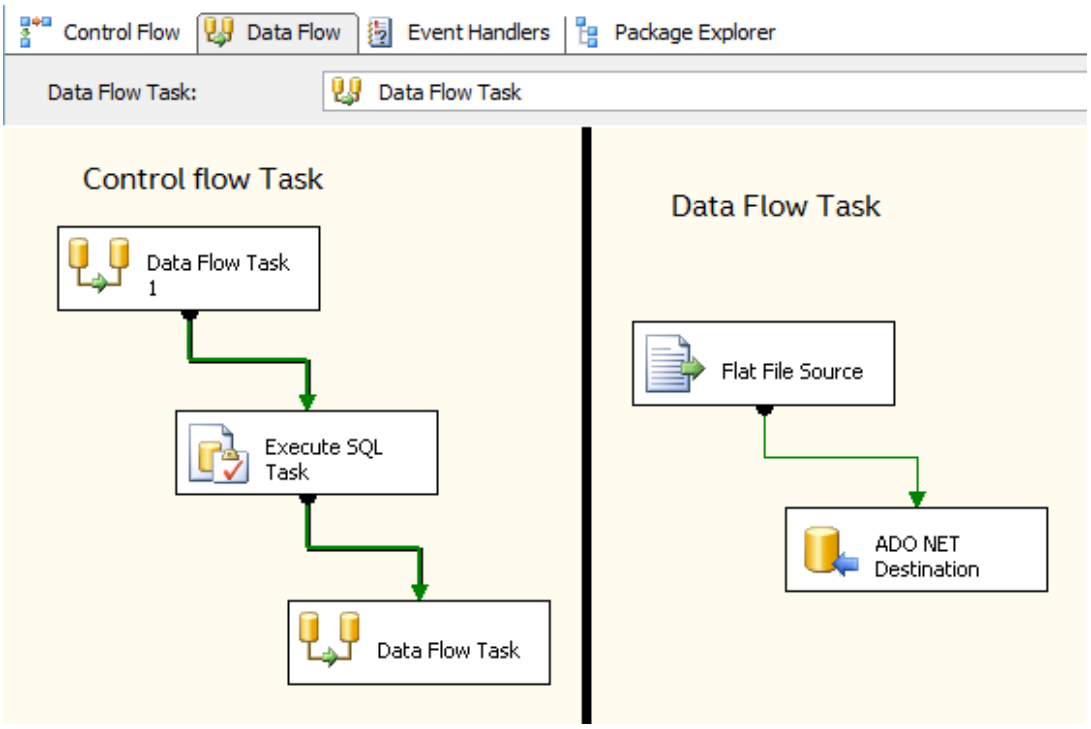
Chapter 9

Working with SSIS and SQL Job Agent

SSIS is used to fetch data from multiple sources and store it at one common destination which is a table in our case. The working methodology has been explained below:

9.1 SSIS Package Creation

Figure 9.1: SSIS Package



- Open Visual Studio 2017, go to file, click New and Select Integration Services

Project.Give name of the project as per Intel's nomenclature.

- Select the data flow task from control flow items.
- Select the data flow source items from data flow task. There can be multiple different source items. Select the data flow transformation items from data flow task. There can be different multiple transformation items. Select the data flow destination item from data flow task.
- As shown in figure 9.1, We selected flat file source and ADO.Net destination. In data flow task we used data flow task then execute SQL task to delete the data fro table and then data flow task. The idea of adding multiple times data flow task task is, if we don't have the file in source folder it will not delete the data from table.
- In the deployment settings, Set the *CreateDeploymentUtility = True*, For generating the *SSIS packages* and *.manifest* file in bin folder.

9.2 Deployment of SSIS Package

- Open the *.manifest* file from the bin directory. This will run the package installation wizard.
- Select the deployment type as SQL server deployment. Enter the Server Details and destination folder[6]

9.3 Creating SQL Job

- As shown in figure 9.2, Create a connection to SQL server and Create a new SQL job by right clicking on the Job directory.
- Enter general details of the job as shown in figure 9.3. Create steps to execute. Select type of step as SQL server integration service package.
- Enter the Server details and Select the SSIS package form the deployment folder
- Now, schedule the job on user defined frequency and time as shown in figure 9.4.

Figure 9.2: SQL Job Wizard

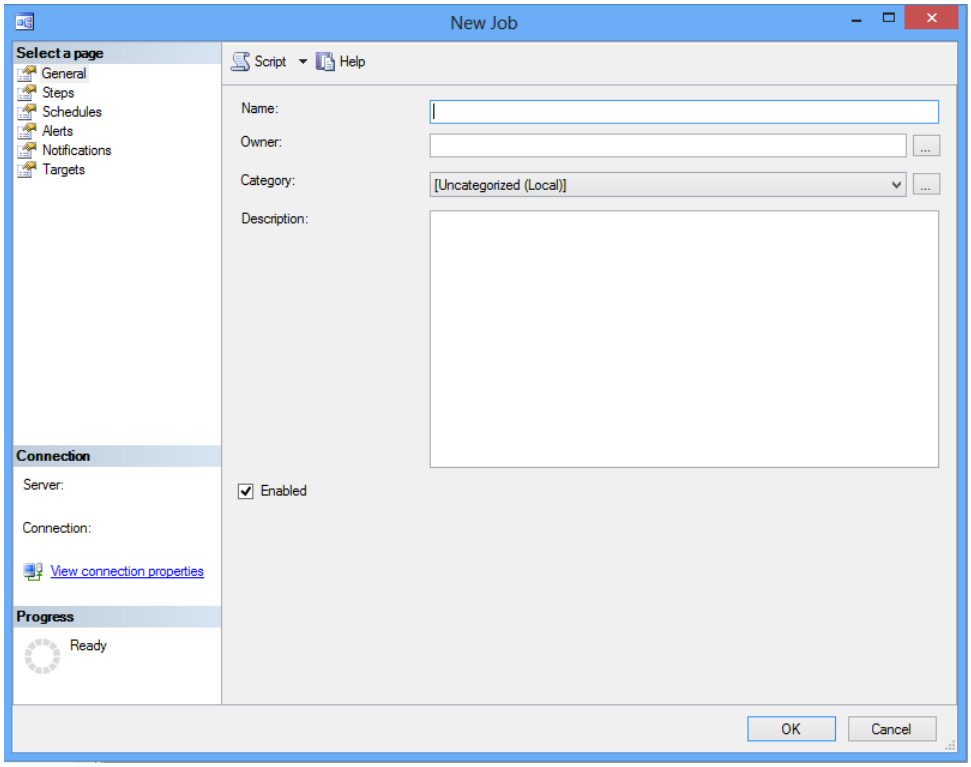


Figure 9.3: Creating Step

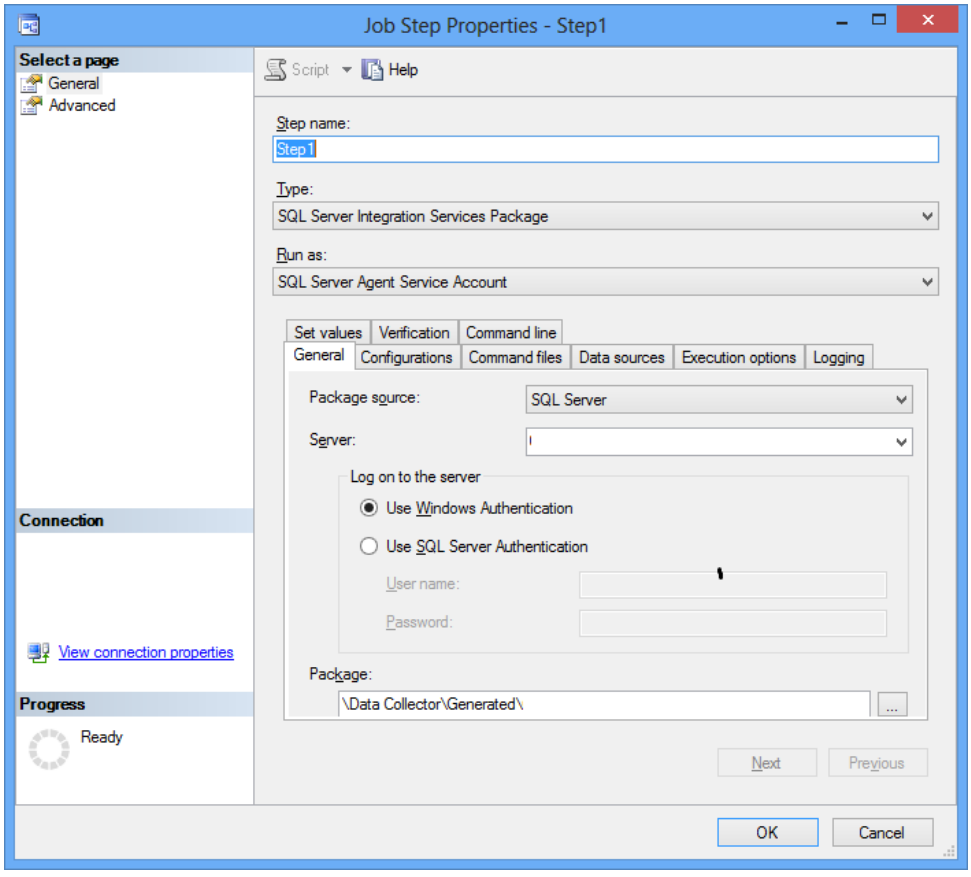
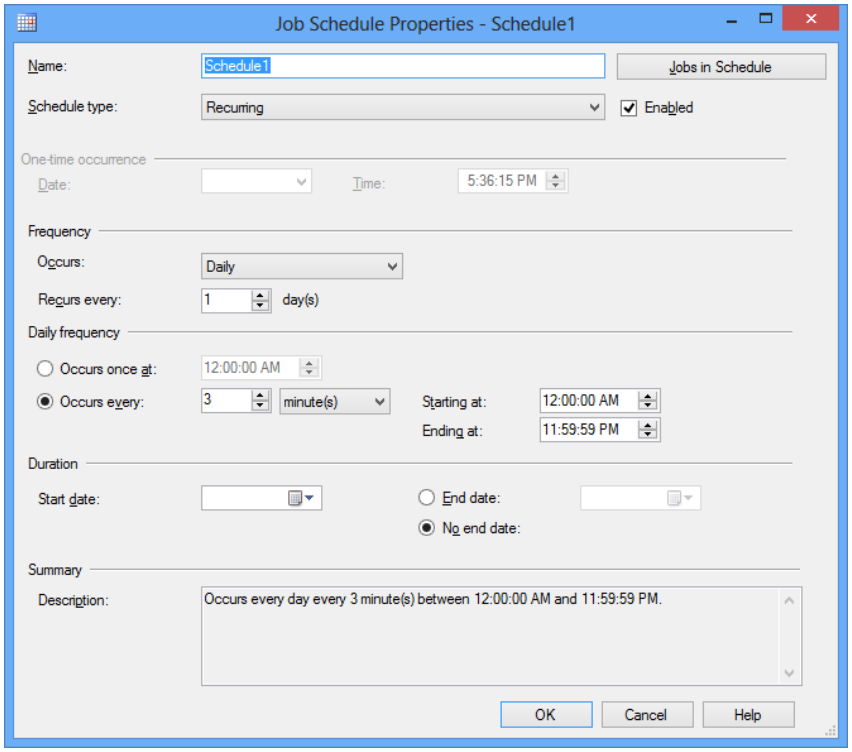
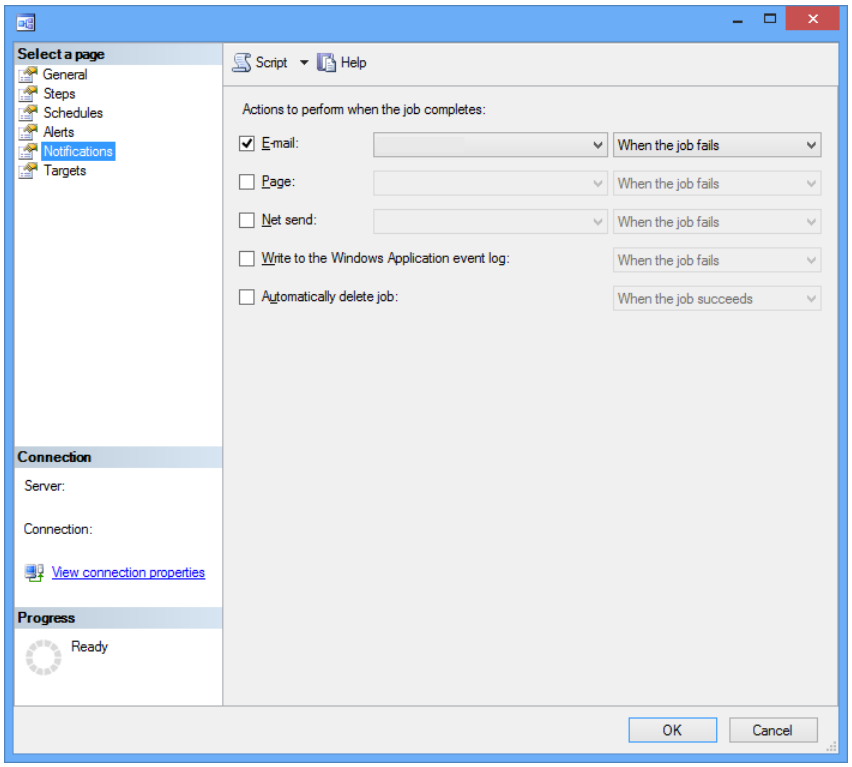


Figure 9.4: Scheduling Job



- Set the email notification to the user when job fails or succeeds,as shown in figure 9.5.

Figure 9.5: Email Notification



Chapter 10

Performance Testing

10.1 Software development performance testing paradigms

It's very important to increase the performance of web applications. To check the performance of web-based application we can use different parameters. According to software prior requirement, we should have to tack care of the performance attribute like scalability, speed, the stability of the software according to the web application's workload. The purpose of Performance Testing is to find functional defects but to eliminate performance bottlenecks in the software or device.

10.2 Application performance parameters

We are here using the Apache JMeter for load testing of web application and servers. Using this tool we can check performance parameters like throughput, the response time of the page, size of the requested page, TTFB(time to first bit), page load, etc.

Figure 10.1: Application Performance Index

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.667	**500 ms**	**1 sec 500 ms**	**Total**
0.500	500 ms	1 sec 500 ms	
0.500	500 ms	1 sec 500 ms	
1.000	500 ms	1 sec 500 ms	

Figure 10.2: Page Requests Summary

Requests Summary

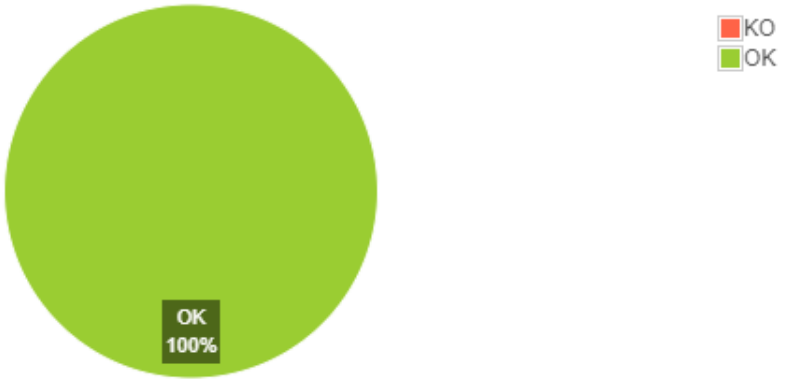


Figure 10.3: Statistics

Statistics

Requests	Executions			Response Times (ms)						Throughput	Network (KB/sec)	
Label	#Samples	KO	Error %	Average	Min	Max	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**3**	**0**	**0.00%**	**784.33**	**490**	**1119**	**1119.00**	**1119.00**	**1119.00**	**1.27**	**820.24**	**0.82**
	1	0	0.00%	744.00	744	744	744.00	744.00	744.00	1.34	93.78	0.86
	1	0	0.00%	1119.00	1119	1119	1119.00	1119.00	1119.00	0.89	910.79	0.58
	1	0	0.00%	490.00	490	490	490.00	490.00	490.00	2.04	1721.54	1.31

Figure 10.4: Response Time Percentiles

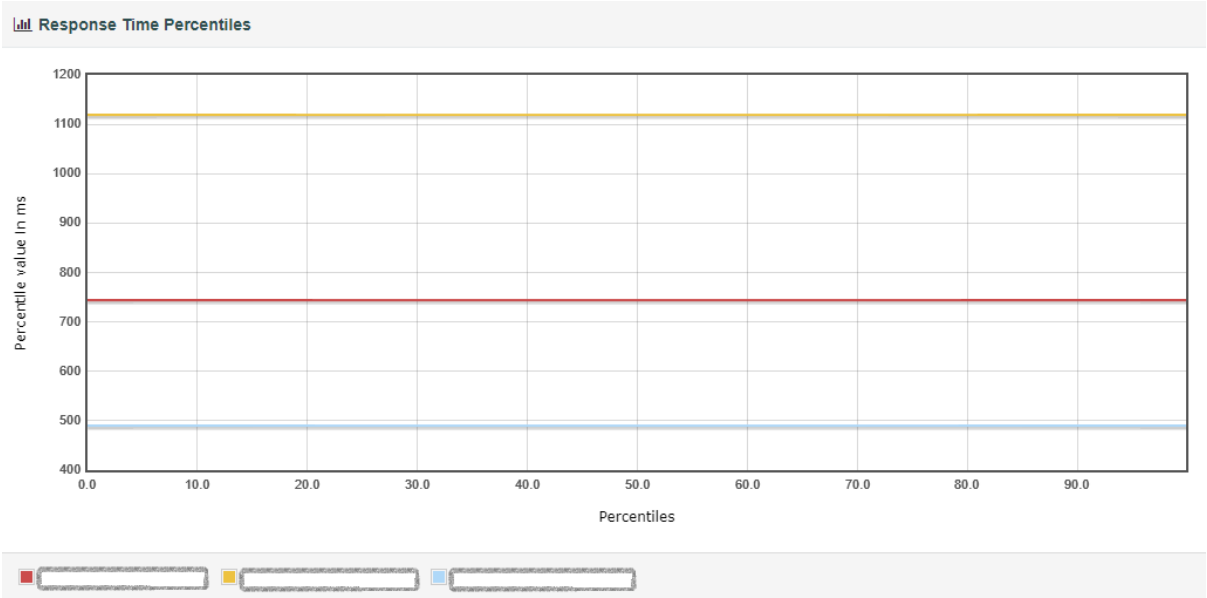


Figure 10.5: Response Time Overview

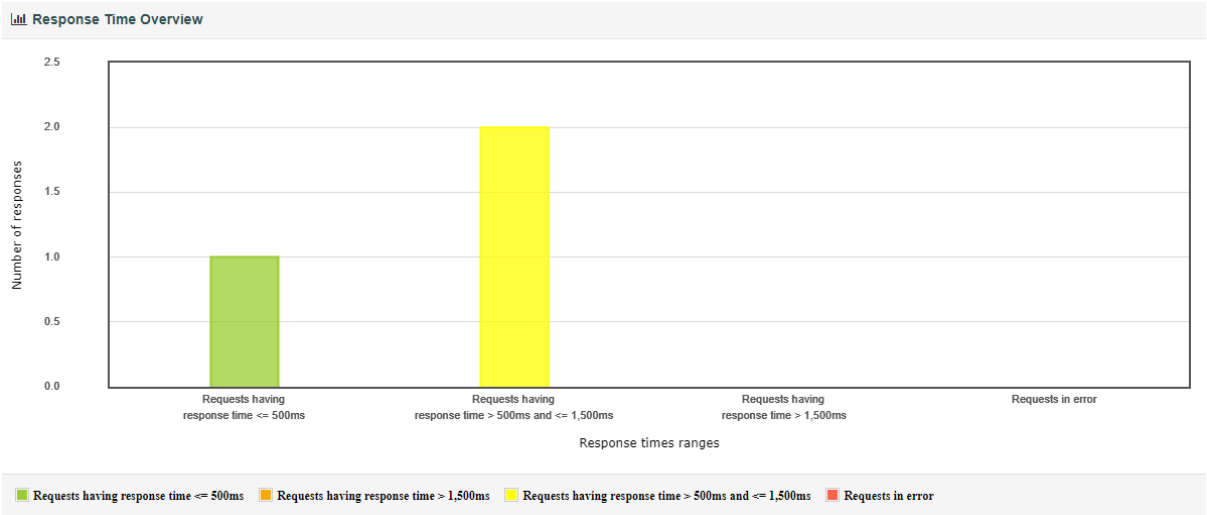


Figure 10.6: Time Vs Threads

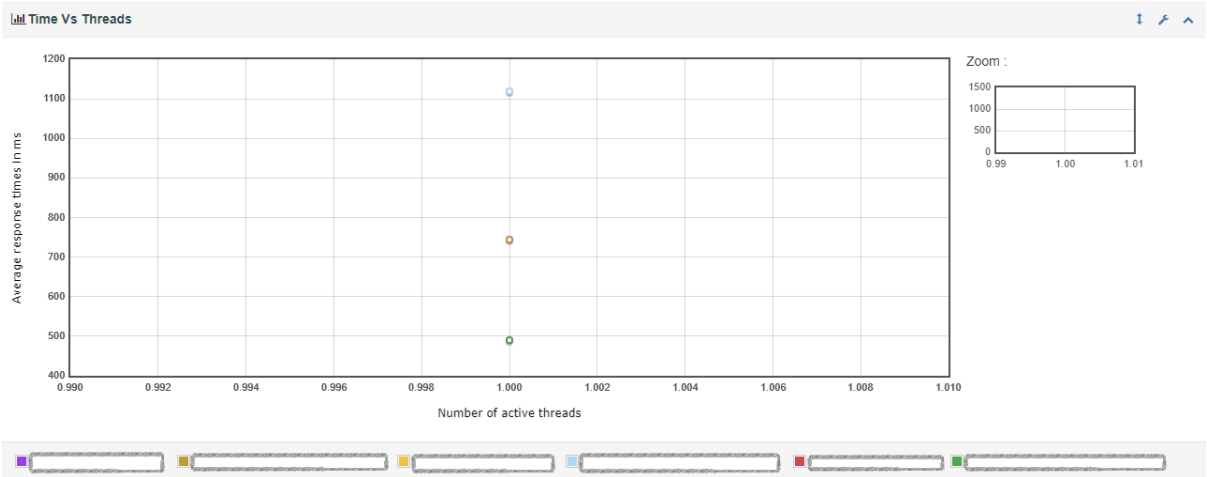
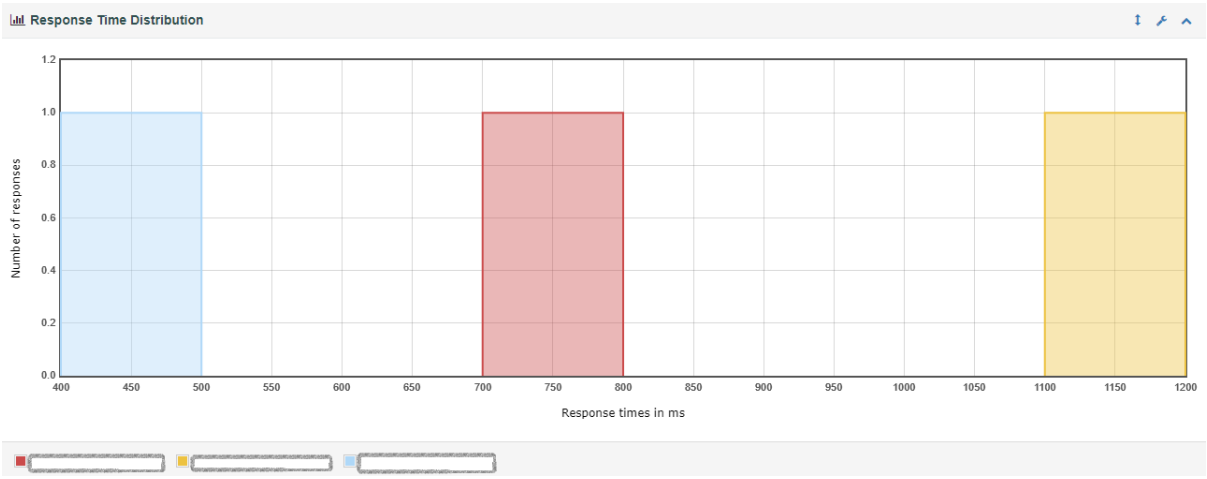


Figure 10.7: Response Time Distribution



Chapter 11

Conclusion

Smart assistant for System Integration and validation is next-generation web-based application, which consists of multiple apps like test-planner, validation generation, recommendation system. This assistance will consistently and directly impact the performance and direction of the Intel Business Group. It drastically reduced the back and forth follow up within the team. It will reduce the no of test cases to be executed on the each platform. So, Using the tool any Intel platform can be released to the client as soon as possible.

11.1 Future Work

Need to implement machine learning algorithm to compute each test case weight and ranks. So, Assistant can easily reduce the no of test cases.

References

[1] https://newsroom.intel.com/news/five-facts-intels-client-computing group/.

[2] D. M. Helge Spieker, Arnaud Gotlieb, M. M. . R. L. for Automatic Test Case Prioritization, and S. in Continuous Integration.

[3] H. V.-B. A. for Cost-Cognizant Test Case Prioritization to Improve the Effectiveness of Regression Testing.

[4] A. Orso and I. G Rothermel. 2014. Software Testing: a Research Travelogue (2000–2014). In Proceedings of the on Future of Software Engineering. ACM, Hyderabad.

[5] O. A. at https://towardsdatascience.com/point-biserial-correlation-with-python f7cd591bd3b1.

[6] S. Online Available at http://technet.microsoft.com/en us/library/ms141026.aspx.