Software User Manual

Version 1.0

CONTINUOUS INTEGRATION TRACEABILITY VISUALIZATION THROUGH HEAT MAP

Prepared By: Mohtashim Abbas Bokhari
Date: August 20, 2016

Supervisors: Dr. Ola Leifler & Dr. Kristian Sandhal

Department of Computer and Information Science-IDA
Linköping University

Contents

About this 3
Document Organization 3
Terminologies & Description 3
1. Introduction 5
1.1 Overview 5
1.2 Applicability 5
1.3 Purpose 5
2. Software Installation 7
2.1 System Requirements 7
2.2 Installation Packages 7
2.3 Running Steps 7
3. How to use software 9

About this

Continuous Integration Traceability Visualization through Heat Map

Software User Manual

Version 1.0

Document Organization

This document is a complete procedure manual, covering all aspects of software of project "*Continuous Integration Traceability Visualization through Heat Map*". It instructs how to install and use the respective software.

This manual is divided into following sections.

Introduction Software Installation How to use software

Terminologies & Description

Name/terminologies	**Description**
Component	A module with many requirements (could be a variant)
Stability of commit	Confidence level of a commit
Aggregated view	Collective results e.g. number of code changes, test results etc.
Individual instances view	Individual code change, traceable graph view
Heat map	Visualization technique

Chapter 01:

INTRODUCTION

1. Introduction

1.1 Overview

This project is a solution to some of the problems of Vehicle Engineering Volvo and Grundfos. Eiffel framework is used in addressing these problems. The project focuses on filtering and visualization of Eiffel data.

1.2 Applicability

Vehicle Engineering Volvo is facing the problem of lacking traceability in requirements, implementation and verification. This project addresses the issue and provides a way to visualize traceable links from implementation back to the requirements.

GrundFos team is facing problems on visualizing status of different commits and their stability* which is being addressed through this project.

1.3 Purpose

This project aims to

- visualize the traceable links mainly between test cases, requirements and implementation
- visualize current status and stability of the commits
- provide an abstract to specific view

* Stability refers to confidence on the number of tests passed showing the progress on different commits

Chapter 02:

SOFTWARE INSTALLATION

2. Software Installation

2.1 System Requirements

System prerequisites are

- Windows: Windows 7 or above / Windows Server 2008 or above
- Processor: Core i3 or above
- Hard drive: 100GB or more
- Installed memory (RAM): 4GB or more
- System type: 32/64 Bit Operating System, x86/x64 based processor

2.2 Installation Packages

- Meteor
- node.js (in case if it is not already installed as a part of meteor)

2.3 Running Steps

1- Download project from the following link.

https://github.com/CI-Visualization/Heatmap-Visualization

2- Open command prompt and write the following.

```
-> 	cd <project_path>
-> 	meteor
```

3- Install relevant packages by writing "npm install <package_name>" in the command prompt.

Chapter 03:

HOW TO USE SOFTWARE

3. How to use software

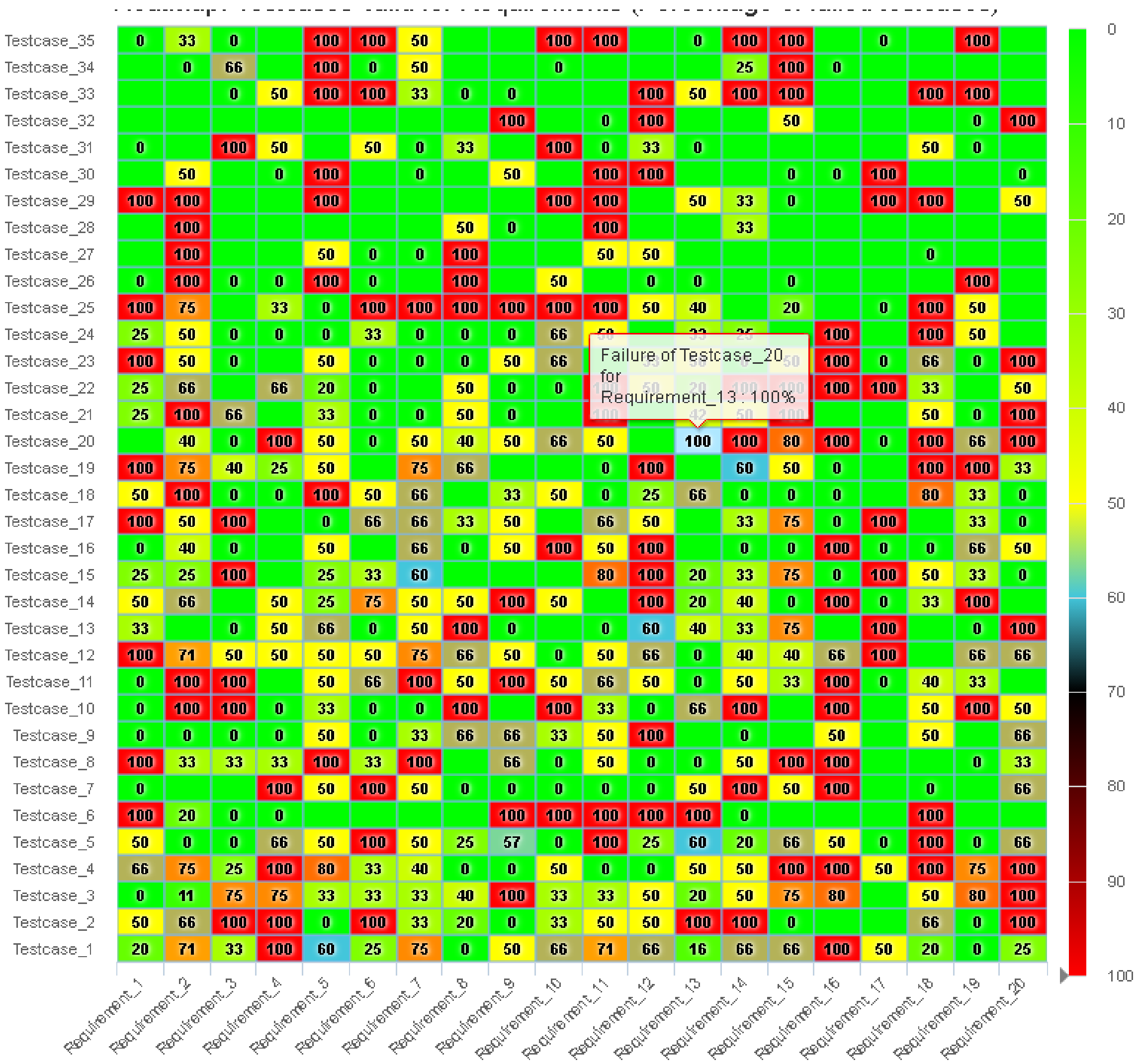
Following are the steps which help the user to run the software mentioned above.

- Type the URL in address bar of browser to run application.

  NOTE: Currently, it is running on http://localhost:3000

- When application is loaded, it provides heat map which is an abstract view of project.

  This heat map view shows user, a relationship between test cases and requirements as shown in Figure 1.

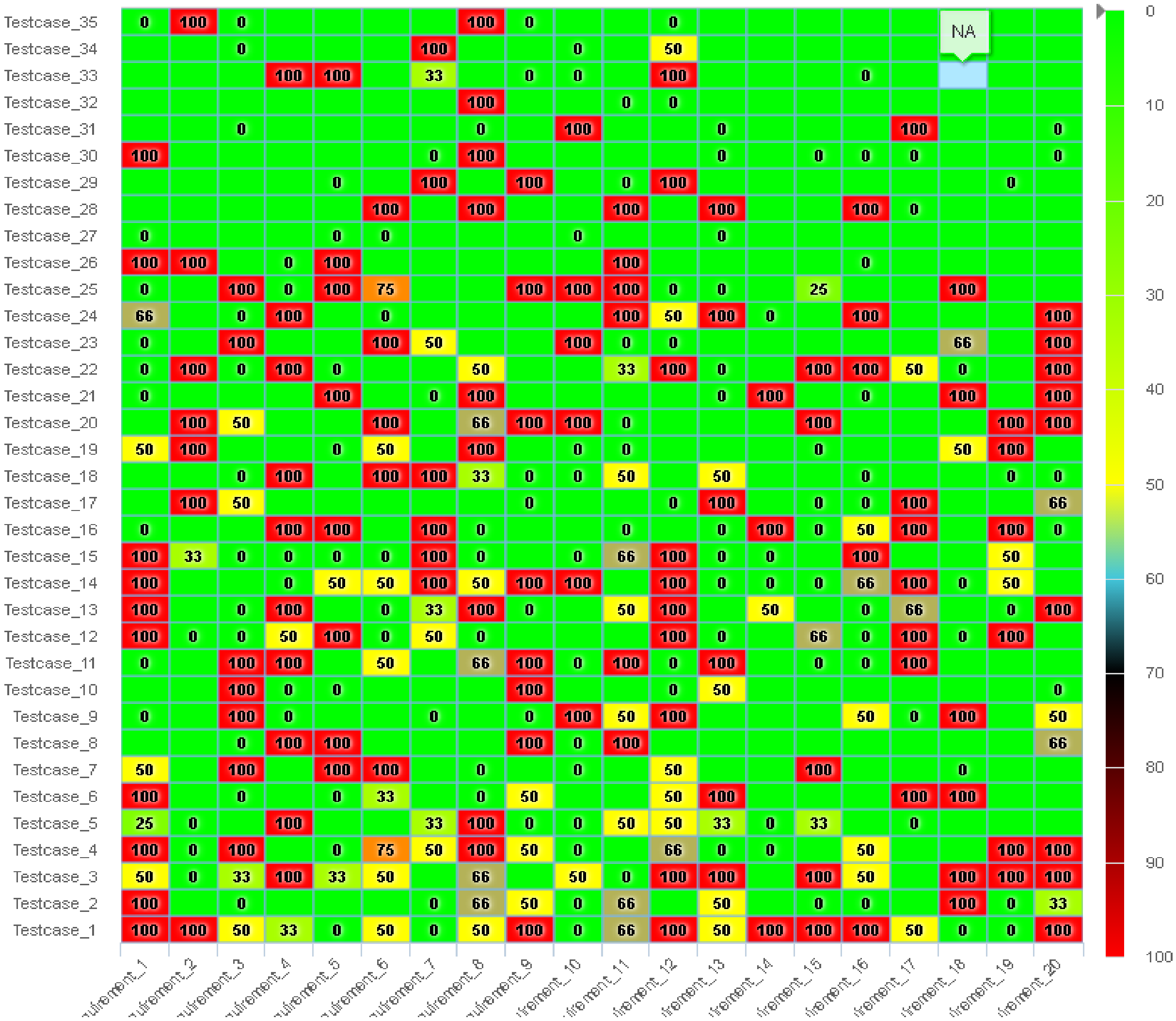


Failure of Testcase_20 for Requirement_13 : 100%

	Requirement_1	Requirement_2	Requirement_3	Requirement_4	Requirement_5	Requirement_6	Requirement_7	Requirement_8	Requirement_9	Requirement_10	Requirement_11	Requirement_12	Requirement_13	Requirement_14	Requirement_15	Requirement_16	Requirement_17	Requirement_18	Requirement_19	Requirement_20
Testcase_35	0	33	0		100	100	50			100	100		0	100	100		0		100	
Testcase_34		0	66		100	0	50			0				25	100	0				
Testcase_33			0	50	100	100	33	0	0			100	50	100	100			100	100	
Testcase_32									100		0	100			50				0	100
Testcase_31	0		100	50		50	0	33		100	0	33	0					50	0	
Testcase_30		50		0	100		0		50		100	100			0	0	100			0
Testcase_29	100	100			100					100	100		50	33	0		100	100		50
Testcase_28		100						50	0		100			33						
Testcase_27		100			50	0	0	100			50	50						0		
Testcase_26	0	100	0	0	100	0		100		50		0	0		0				100	
Testcase_25	100	75		33	0	100	100	100	100	100	100	50	40		20		0	100	50	
Testcase_24	25	50	0	0	0	33	0	0	0	66	50		33	25		100		100	50	
Testcase_23	100	50	0		50	0	0	0	50	66	[illegible]	33	50	0	50	100	0	66	0	100
Testcase_22	25	66		66	20	0		50	0	0	100	50	20	100	100	100	100	33		50
Testcase_21	25	100	66		33	0	0	50	0		100		42	50	100			50	0	100
Testcase_20		40	0	100	50	0	50	40	50	66	50		100	100	80	100	0	100	66	100
Testcase_19	100	75	40	25	50		75	66			0	100		60	50	0		100	100	33
Testcase_18	50	100	0	0	100	50	66		33	50	0	25	66	0	0	0		80	33	0
Testcase_17	100	50	100		0	66	66	33	50		66	50		33	75	0	100		33	0
Testcase_16	0	40	0		50		66	0	50	100	50	100		0	0	100	0	0	66	50
Testcase_15	25	25	100		25	33	60				80	100	20	33	75	0	100	50	33	0
Testcase_14	50	66		50	25	75	50	50	100	50		100	20	40	0	100	0	33	100	
Testcase_13	33		0	50	66	0	50	100	0		0	60	40	33	75		100		0	100
Testcase_12	100	71	50	50	50	50	75	66	50	0	50	66	0	40	40	66	100		66	66
Testcase_11	0	100	100		50	66	100	50	100	50	66	50	0	50	33	100	0	40	33	
Testcase_10	0	100	100	0	33	0	0	100		100	33	0	66	100		100		50	100	50
Testcase_9	0	0	0	0	50	0	33	66	66	33	50	100		0		50		50		66
Testcase_8	100	33	33	33	100	33	100		66	0	50	0	0	50	100	100			0	33
Testcase_7	0			100	50	100	50	0	0	0	0	0	50	100	50	100		0		66
Testcase_6	100	20	0	0					100	100	100	100	100	0				100		
Testcase_5	50	0	0	66	50	100	50	25	57	0	100	25	60	20	66	50	0	100	0	66
Testcase_4	66	75	25	100	80	33	40	0	0	50	0	0	50	50	100	100	50	100	75	100
Testcase_3	0	11	75	75	33	33	33	40	100	33	33	50	20	50	75	80		50	80	100
Testcase_2	50	66	100	100	0	100	33	20	0	33	50	50	100	100	0			66	0	100
Testcase_1	20	71	33	100	60	25	75	0	50	66	71	66	16	66	66	100	50	20	0	25

Figure 1

In figure 1, x-axis represents all the requirements, y-axis displays all the test cases and color represents percentage failure of a test case. Percentage ranges from 0-100% and every result has its color in heat map. Red is for critical case which indicates that a test case fails as many times as it is applied on the code changes, according to a certain requirement whereas green color depicts passed test cases.



	uirement_1	uirement_2	uirement_3	uirement_4	uirement_5	uirement_6	uirement_7	uirement_8	uirement_9	irement_10	irement_11	irement_12	irement_13	irement_14	irement_15	irement_16	irement_17	irement_18	irement_19	irement_20
Testcase_35	0	100	0					100	0			0								
Testcase_34			0				100			0		50								
Testcase_33				100	100		33		0	0		100				0				
Testcase_32								100			0	0								
Testcase_31			0					0		100			0				100			0
Testcase_30	100						0	100					0		0	0	0			0
Testcase_29					0		100		100		0	100							0	
Testcase_28						100		100			100		100			100	0			
Testcase_27	0				0	0				0			0							
Testcase_26	100	100		0	100						100					0				
Testcase_25	0		100	0	100	75			100	100	100	0	0		25			100		
Testcase_24	66		0	100		0					100	50	100	0		100				100
Testcase_23	0		100			100	50			100	0	0						66		100
Testcase_22	0	100	0	100	0			50			33	100	0		100	100	50	0		100
Testcase_21	0				100		0	100					0	100		0		100		100
Testcase_20		100	50			100		66	100	100	0				100				100	100
Testcase_19	50	100			0	50		100		0	0				0			50	100	
Testcase_18			0	100		100	100	33	0	0	50		50			0			0	0
Testcase_17		100	50						0			0	100		0	0	100			66
Testcase_16	0			100	100		100	0			0		0	100	0	50	100		100	0
Testcase_15	100	33	0	0	0	0	100	0		0	66	100	0	0		100			50	
Testcase_14	100			0	50	50	100	50	100	100		100	0	0	0	66	100	0	50	
Testcase_13	100		0	100		0	33	100	0		50	100		50		0	66		0	100
Testcase_12	100	0	0	50	100	0	50	0				100	0		66	0	100	0	100	
Testcase_11	0		100	100		50		66	100	0	100	0	100		0	0	100			
Testcase_10			100	0	0				100			0	50							0
Testcase_9	0		100	0			0		0	100	50	100				50	0	100		50
Testcase_8			0	100	100				100	0	100									66
Testcase_7	50		100		100	100		0		0		50			100			0		
Testcase_6	100		0		0	33		0	50			50	100				100	100		
Testcase_5	25	0		100			33	100	0	0	50	50	33	0	33		0			
Testcase_4	100	0	100		0	75	50	100	50	0		66	0	0		50			100	100
Testcase_3	50	0	33	100	33	50		66		50	0	100	100		100	50		100	100	100
Testcase_2	100		0				0	66	50	0	66		50		0	0		100	0	33
Testcase_1	100	100	50	33	0	50	0	50	100	0	66	100	50	100	100	100	50	0	0	100

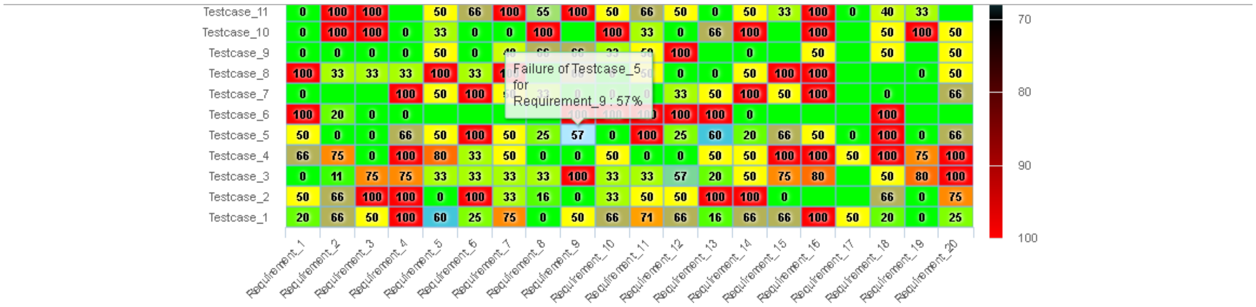
Figure 2

- In Figure 2, an empty cell in the heat map denotes that test case doesn’t apply on the code changes which belong to concerned requirement. If a given test case is not relevant, it shows not applied (NA), otherwise it displays testing results to the user.

Colors: Red- critical
Green- normal

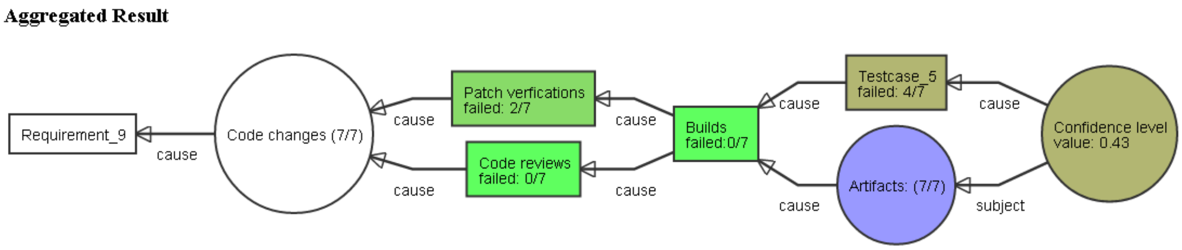
NOTE: This heat map is an abstract view of all the test cases and requirements. User clicks on the heat map view to get information about code changes for any requirement.

- When user clicks on map, system opens an aggregated view as illustrated in Figure 3. It provides information about number of times the test case was performed on different code changes, representing a particular requirement. It also provides information about total number of code changes, reviews, builds, tests, pass/fail information and confidence level (metric for indicating stability of changes) for the selected requirement.



	Requirement_1	Requirement_2	Requirement_3	Requirement_4	Requirement_5	Requirement_6	Requirement_7	Requirement_8	Requirement_9	Requirement_10	Requirement_11	Requirement_12	Requirement_13	Requirement_14	Requirement_15	Requirement_16	Requirement_17	Requirement_18	Requirement_19	Requirement_20
Testcase_11	0	100	100		50	66	100	55	100	50	66	50	0	50	33	100	0	40	33	
Testcase_10	0	100	100	0	33	0	0	100		100	33	0	66	100		100		50	100	50
Testcase_9	0	0	0	0	50	0	40	66	66	33	50	100		0		50		50		50
Testcase_8	100	33	33	33	100	33	100	[illegible]	66	0	50	0	0	50	100	100			0	50
Testcase_7	0			100	50	100	50	33	0	0	0	33	50	100	50	100		0		66
Testcase_6	100	20	0	0					100	100	100	100	100	0				100		
Testcase_5	50	0	0	66	50	100	50	25	57	0	100	25	60	20	66	50	0	100	0	66
Testcase_4	66	75	0	100	80	33	50	0	0	50	0	0	50	50	100	100	50	100	75	100
Testcase_3	0	11	75	75	33	33	33	33	100	33	33	57	20	50	75	80		50	80	100
Testcase_2	50	66	100	100	0	100	33	16	0	33	50	50	100	100	0			66	0	75
Testcase_1	20	66	50	100	60	25	75	0	50	66	71	66	16	66	66	100	50	20	0	25

Failure of Testcase_5 for Requirement_9 : 57%



Aggregated Result

Figure 3

- User scrolls down from aggregated view to check individual events view. Here, selected requirement can be seen and traced as individual events as shown in Figure 4. This view provides further information about the code changes, verifications, reviews, builds, tests, testing date, results and confidence level of every single change for any requirement. These individual events refer to the same requirement and test case as clicked by the user.

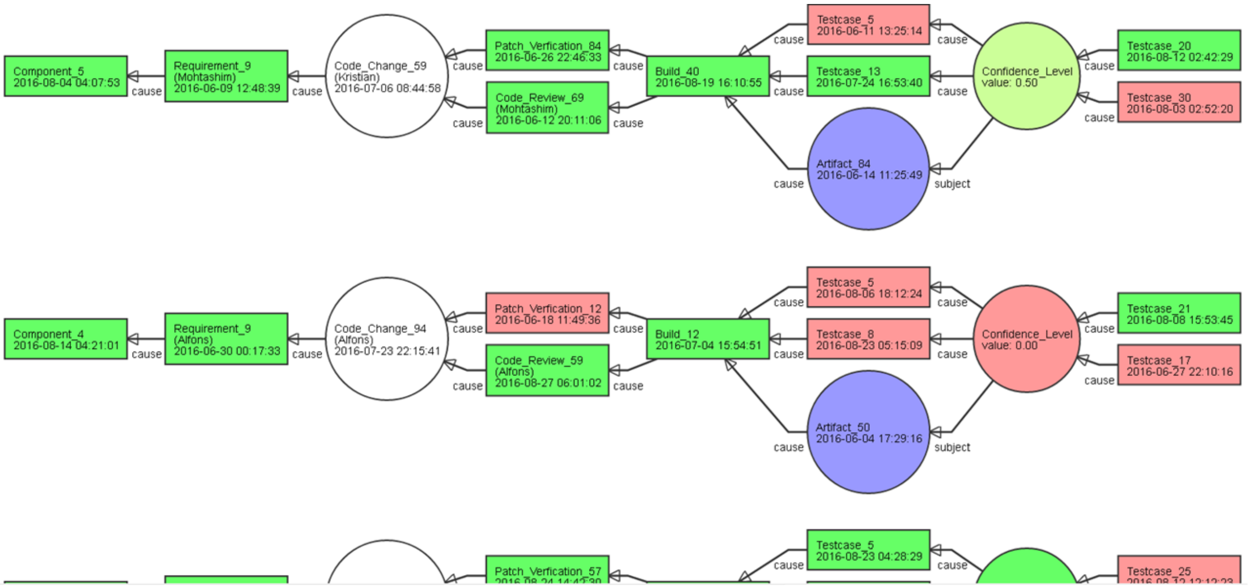
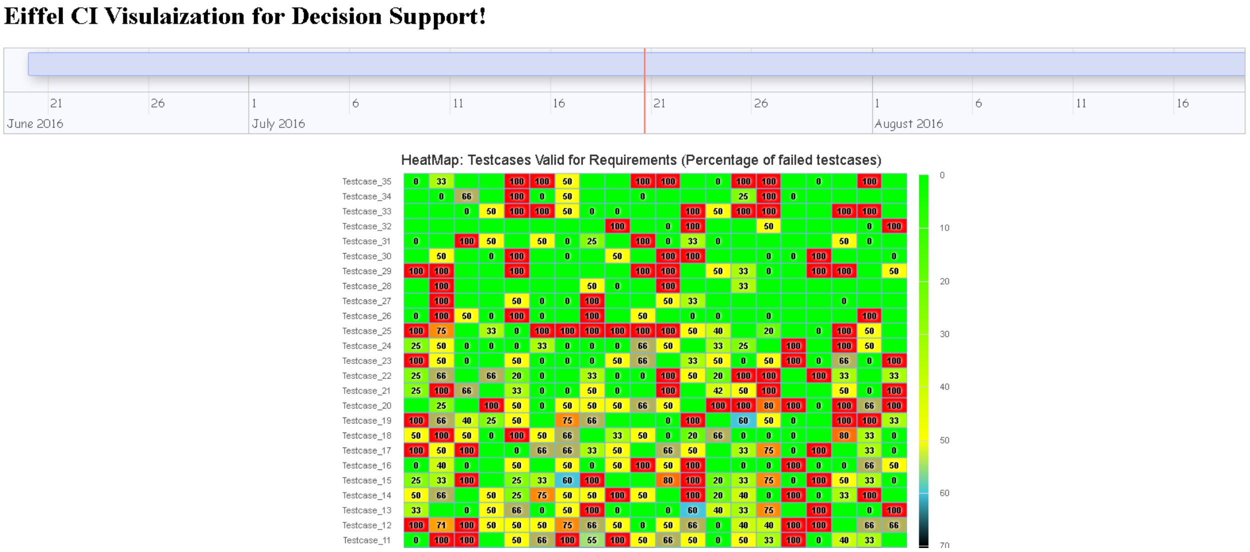


Figure 4

- User selects *Date* from the timeline by moving it sideways and system visualizes the Eiffel continuous integration data accordingly as shown in Figure 5.


Eiffel CI Visulaization for Decision Support!

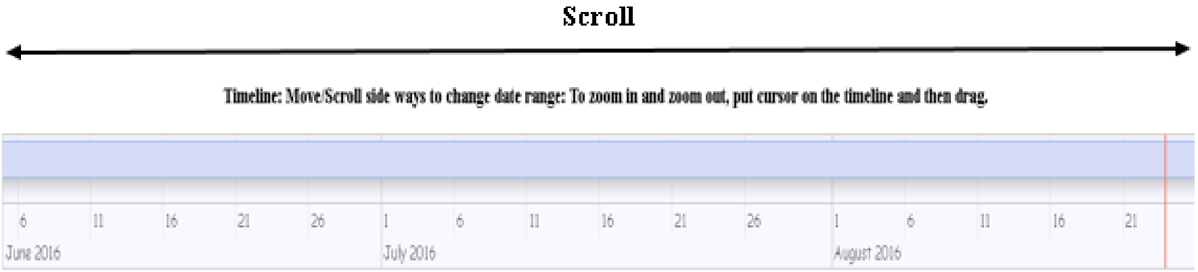
HeatMap: Testcases Valid for Requirements (Percentage of failed testcases)

Testcase_35	0	33			100	100	50			100	100		0	100	100		0		100	
Testcase_34		0	66		100	0	50			0				25	100	0				
Testcase_33			0	50	100	100	50	0	0			100	50	100	100			100	100	
Testcase_32									100		0	100			50				0	100
Testcase_31	0		100	50		50	0	25		100	0	33	0					50	0	
Testcase_30		50		0	100		0		50		100	100			0	0	100			0
Testcase_29	100	100			100					100	100		50	33	0		100	100		50
Testcase_28		100						50	0		100			33						
Testcase_27		100			50	0	0	100			50	33						0		
Testcase_26	0	100	50	0	100	0		100		50		0	0		0				100	
Testcase_25	100	75		33	0	100	100	100	100	100	100	50	40		20		0	100	50	
Testcase_24	25	50	0	0	0	33	0	0	0	66	50		33	25		100		100	50	
Testcase_23	100	50	0		50	0	0	0	50	66		33	50	0	50	100	0	66	0	100
Testcase_22	25	66		66	20	0		33	0	0	100	50	20	100	100		100	33		33
Testcase_21	25	100	66		33	0	0	50	0		100		42	50	100			50	0	100
Testcase_20		25		100	50	0	50	50	50	66	50		100	100	80	100	0	100	66	100
Testcase_19	100	66	40	25	50		75	66			0	100		60	50	0		100	100	33
Testcase_18	50	100	50	0	100	50	66		33	50	0	20	66	0	0	0		80	33	0
Testcase_17	100	50	100		0	66	66	33	50		66	50		33	75	0	100		33	0
Testcase_16	0	40	0		50		50	0	50	100	50	100		0	0	100	0	0	66	50
Testcase_15	25	33	100		25	33	60	100			80	100	20	33	75	0	100	50	33	0
Testcase_14	50	66		50	25	75	50	50	100	50		100	20	40	0	100	0	33	100	
Testcase_13	33		0	50	66	0	50	100	0		0	60	40	33	75		100		0	100
Testcase_12	100	71	100	50	50	50	75	66	50	0	50	66	0	40	40	100	100		66	66
Testcase_11	0	100	100		50	66	100	55	100	50	66	50	0	50	33	100	0	40	33	

Figure 5

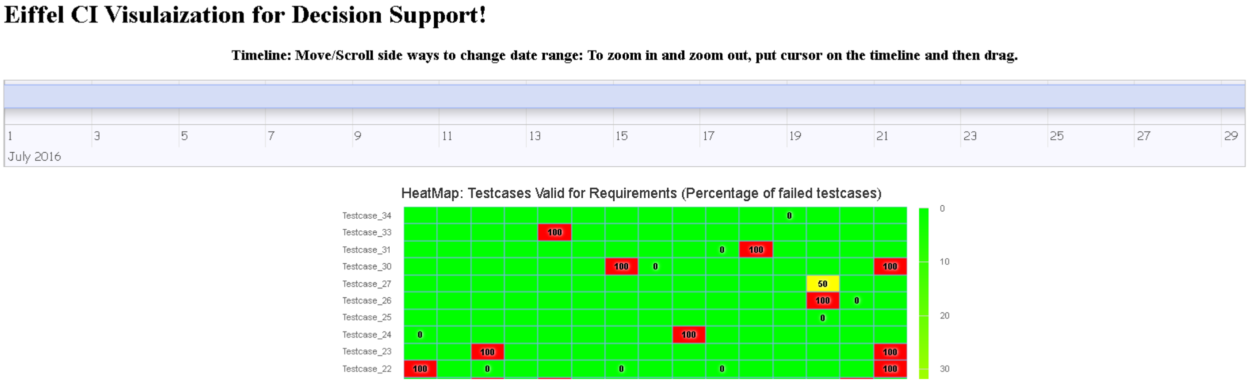
- Timeline basically provides user facility to view CI Eiffel data during any period of time. Its functions are stated below:
  a. It provides user the ability to select date by scrolling the timeline sideways as shown in figure 6.
  b. It provides user the ability to zoom in and zoom out to monitor CI Eiffel data in daily, weekly, monthly and yearly view as shown in figure 7.

NOTE: To change date range scroll timeline sideways; to zoom in and zoom out, put cursor on the timeline and then drag.



Timeline: Move/Scroll side ways to change date range: To zoom in and zoom out, put cursor on the timeline and then drag.

Figure. 6

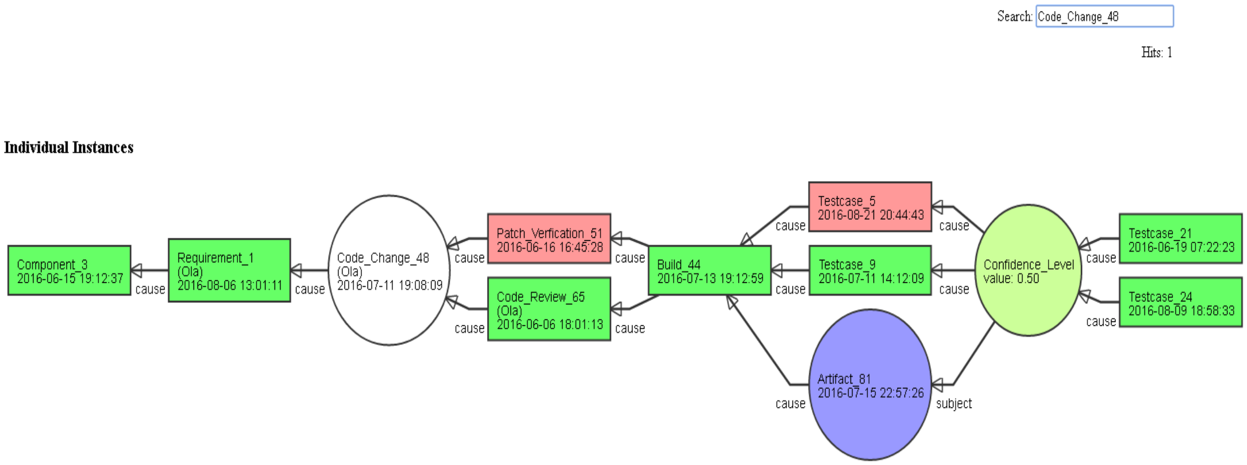


Eiffel CI Visulaization for Decision Support!

Timeline: Move/Scroll side ways to change date range: To zoom in and zoom out, put cursor on the timeline and then drag.

Figure. 7

- Eiffel events can be selected in any period of time through the timeline. A user can easily filter any test case/ requirement/ review/ code changes/ developer/ build/ component etc. by writing *ID* in the search bar as shown in Figure 8& 9.

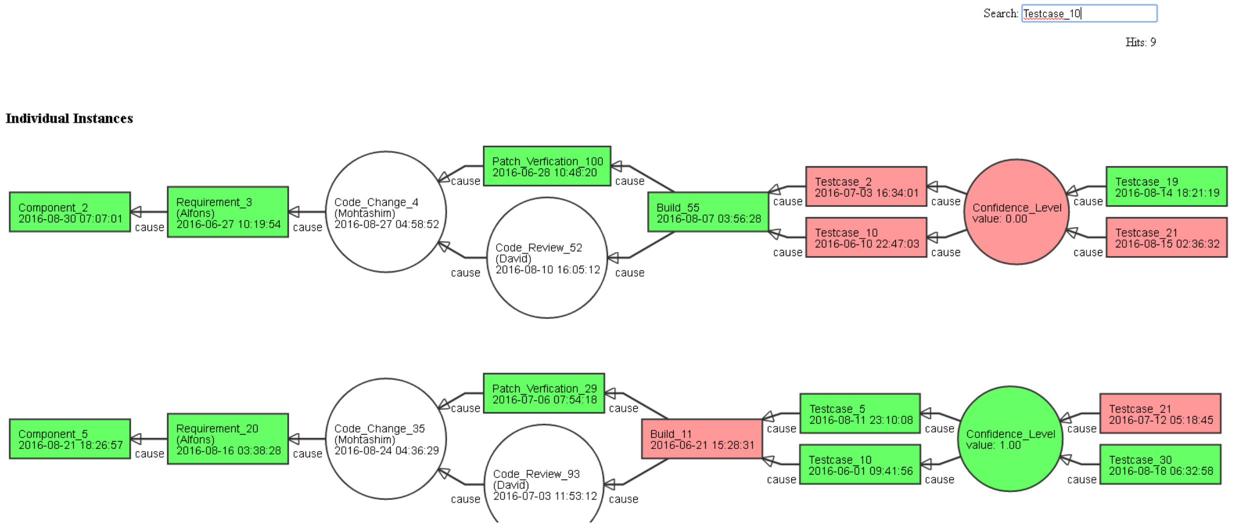


Search: Code_Change_48

Hits: 1

Individual Instances

Figure 8



Search: Testcase_10

Hits: 9

Individual Instances

Figure 9