σ

Journal of Engineering and Natural Sciences
Mühendislik ve Fen Bilimleri Dergisi

**Sigma 32,
10-22,
2014**

Research Article / Araştırma Makalesi

INVESTIGATION OF TEST AUTOMATION DATA USING STATISTICAL METHODS

Muhammed Maruf ÖZTÜRK[*], Ahmet ZENGİN

Sakarya University, Faculty of Technology, Computer Engineering Department, Esentepe-SAKARYA

Received/Geliş: 16.08.2013 Revised/Düzeltme: 24.10.2013 Accepted/Kabul: 05.12.2013

ABSTRACT

As the scale of software projects increase, the time allocated to development of software increases that testing of software quality is becoming important depending of this situation. Software test teams take some responsibilities to reduce time of project development and make particular planning to shorten completion of project as well as the time of maintenance. At this point of view, various automated test tools are widely used to reduce project effort and time. The most common area of using automated test is web based software. Thanks to automated tests, errors can be detected which available in web based software and all transactions are saved. Owing to running of these test tools on the basis of object oriented languages and provided libraries that enable to user controlling these tools are considered as the reason of selection. This study presents a framework that provide controlling of test tools and analyzing test data statistically. The innovation of statistical analysis which performed on software test area has been depicted.

Keywords: Software testing, automated test, statistical analysis.

TEST OTOMASYONU VERİLERİNİN İSTATİSTİKSEL YÖNTEMLERLE İNCELENMESİ

ÖZET

Yazılım projelerinin ölçekleri büyüdükçe yazılım geliştirmeye ayrılan zaman artmakta buna bağlı olarak da yazılım kalitesinin test edilmesi önemli hale gelmektedir. Proje geliştirme zamanlarını düşürmek için yazılım test ekipleri projelerde görev almakta ve hem proje üretimi hem de bakıma ayrılan zamanı azaltmaya yönelik planlamalar yapılmaktadır. Bu noktada teste harcanan zamanı ve çabayı azaltmak için çeşitli otomatik test araçları yaygın bir şekilde kullanılmaktadır. Otomatik testlerin en yaygın kullanıldığı alanlardan biri de web tabanlı yazılımlardır. Otomatikleştirilmiş testlerle web tabanlı yazılımlardaki hatalar tespit edilebilmekte ve yapılan test işlemleri kaydedilmektedir. Bu test araçlarının nesne yönelimli programlama dilleri ile uyumlu çalışması ve araçların kontrol edilmesini sağlayan kütüphane desteğinin sunulması tercih nedeni olmaktadır. Bu çalışmada otomatik test araçlarının yönetimini sağlayan ve test verilerinin istatistiksel analizini sağlayan bir çerçeve sunulmuştur. Yazılım test çalışmalarında istatistiksel analizlerin test yönetimine getirdiği yenilik vurgulanmıştır.

Anahtar Sözcükler: Yazılım testi, otomatik test, istatistiksel analiz.

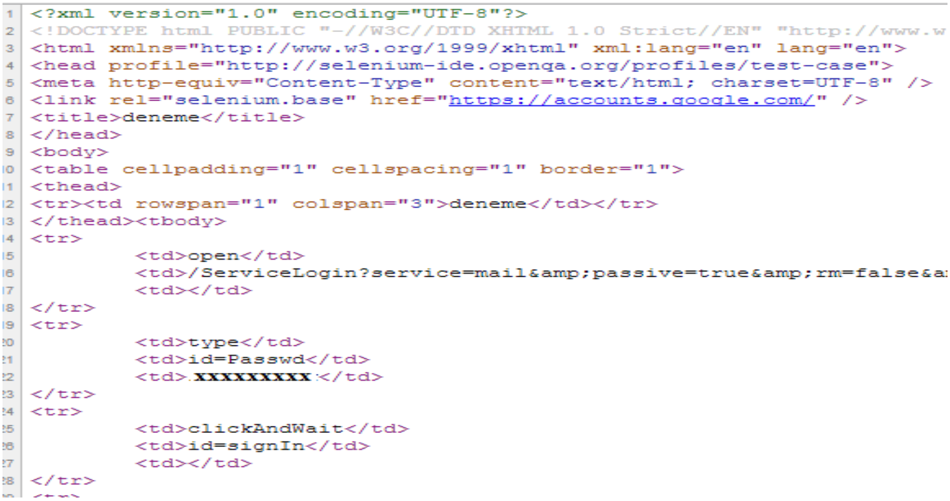
[*] Corresponding Author/Sorumlu Yazar: e-mail/e-ileti: muhammedozturk@sakarya.edu.tr, tel: (264) 295 69 09

1. INTRODUCTION

Test-driven development has become more effective programming idea, known as one of the agile methods, than other ideas in the world [1]. Test-driven development aims to both improvement the validation of codes and reducing defect rates [2]. Initially test techniques used by a minor group as superior skills have become to be known among development teams. The selection of which technique for a specific goal is more important than the implementation of test techniques [3]. Test-driven development also requires the automation of tests. Automated test tools not only facilitates the preparation of test cases but also presents various test report formats [4], [5]. Automated test software can save performed tests thus the comparison of test results are possible. These tools are preferred according to some features such as usability, language and library support. Automated test tools are used based on determined goals such as functional and acceptance tests and these tools require a special talent and information [6]. Developed control software aims to vanish these constraints and it's superior aspects are stressed presenting analysis results. Developed framework runs on the basis of Selenium and Fitnesse automated test tools. With the management of these test functions, the importance of using statistical methods for test data is discussed.

Selenium [7], [8] is an automated test library that automate web test applications which provide management GUI test. This tool has been produced by ThroughtWorks. Selenium can be managed by various programming languages through library support. The commands sent by CPU are processed aftermath the test results are returned [9]. Tests are run on the browser via Selenium IDE plugin. Functional aspects are more considered than code structure when Selenium is to be run. So this tool is suitable for unit test and GUI test. Because the inputs are entered to system then determined outputs are expected also the accuracy of system functions are verified [10]. Performed test are saved as xhtml file seen in Figure 1.

Commands are described as HTML table and values are given to targeted elements. Commands used for web element test are seen in Chart 1. The main advantage of Selenium test tool is that recorded test script can be converted to Java or C# programming language codes. An hardship is encountered when the page structure changes, test script changing is also needed. This hardship increases the test cost.



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head profile="http://selenium-ide.openqa.org/profiles/test-case">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link rel="selenium.base" href="https://accounts.google.com/" />
<title>deneme</title>
</head>
<body>
<table cellpadding="1" cellspacing="1" border="1">
<thead>
<tr><td rowspan="1" colspan="3">deneme</td></tr>
</thead><tbody>
<tr>
        <td>open</td>
        <td>/ServiceLogin?service=mail&amp;passive=true&amp;rm=false&a
        <td></td>
</tr>
<tr>
        <td>type</td>
        <td>id=Passwd</td>
        <td>XXXXXXXXXX</td>
</tr>
<tr>
        <td>clickAndWait</td>
        <td>id=signIn</td>
        <td></td>
</tr>
```

Figure 1. Selenium xhtml test code

Chart 1. Common Selenium functions [11]

Command Name	Description
type(locator, value)	Fill web field
select(selectLocator, opt.Locator)	Selection of dropdownlist
click(locator)	Execution of click()
xxxAndWait	wait after function
verifyXXX(locator, pattern)	Matching control
assert	Stop test after failed result

Fitnesse an open source software known as one of the acceptance test tools. Thanks to Fitnesse test cases can be written using various programming languages such as Java, .Net or Pyhton. The comparison of actual and expected result test can be performed using developed classes and functional structures. Test tables can be arranged based on Excel, Word or any HTML editor that facilitates writing of the tests.

2. AUTOMATED TEST ARCHITECTURES

When designing test architectures some purposes are determined to reduce test cost and validation process also to improve the quality of software product. Automated tests assist to detect defect prone parts of programs [12]. The main reason of the need for automated test is to make tests repeated. Iteration of test harness increases error detection rate. Besides, automated tests can determine whether the system is deterministic. If different test results are obtained using the same test scenario we can say that the system is not deterministic. One point should be determined in automated test harness beforehand that which type of tests will be given priority. Unit test rate should be more than acceptance and GUI tests in a well-designed automated test [13]. To manage the quality of a software product, it's all of properties should be tested. If the product is a large-scale software in such a case this software test process will be very hard to complete. In addition, this situation will reveal a pressure on test team that test processes should be completed in a shorter period. All these difficulties have been overcome through the facilities of automated test approach. The advantages of automated test approach can be expressed as follows:

— Working time is shortened: Automated test tools complete test cases in a shorter time thus more test scenario can be run in a less time.

— Don't require an intervention: Facilitates the regression test.

— Repeatable: For instance test process can be performed on different browsers with different peripherals.

— Reusable: Completed test scripts can be converted to various programming languages that enable to us portable tests.

— Consistency: Test manager could make a mistake on the step count of test for example test could be completed in 9 steps instead of 10 steps but this type of defect is not encountered in automated test environment.

— Scope: Because of the absence of time restriction, more test scenario can be run in the same period.

— Resource efficient approach: Regression test uses fewer resource that reduce test cost.

3. RELATED WORKS

3.1. Management of Automated Test

Catelani et al. [12] have proposed a remedy using automated test that extends the scope of test plan. Proposed remedy has improved the quality and the reliability of the software. The software that under a certain level of stress has been simulated, sequential processes have been accelerated four times with automated test approach when comparing with manual test approach. In Shahamiri et al.'s work [4] a novel automated test framework has been presented. Developed framework has revealed system defects with a certain rate among 91.17%-98.26%. The results of this study has been evaluated using artificial neural networks and mutation test.

Löffler and his friends [15] has been developed a test model based on Scrum. In this model, the test tables of tests are automatically generated which performed using Selenium. The effectiveness of the developed model has been depicted on an industrial application. Another experimental industrial application [16] has explained the hardship of cost reduction. To address this problem test automation systems should be designed with agile techniques. Emery's work [17] performed in 2009 has determined the restrictions of test automation, some suggestions have been presented related to this problem. In Xe's work [18] the support of symbolic execution has been mentioned in automated test, one can be concluded from this work that tests including feedback improve the automation of systems. Jureczko and Mlynarski's work [19] has compared acceptance test tools. In this comparison, Fitnesse and Proven! have been evaluated the most effective tools for database interaction. When considering all these studies, a web based control tool is needed to facilitate the management of acceptance and GUI tests.

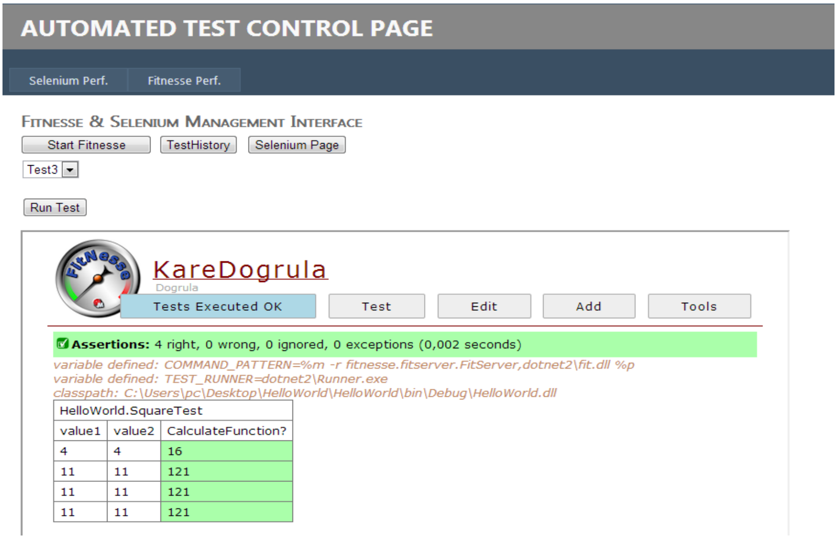
3.2. Software Testing Studies based on Statistic

Pan et al.'s work [14] has presented a framework based on statistic. Developed web based framework has reached a significant success to improve processes. In another work [21] related to statistic, defects have been eliminated using functional validation up to 90%. To software quality improvement Redzic and Baik [22] have investigated Six Sigma methodology as a phase under the measurement phase. Various works are available in literature intended for the verification of software quality process. Galinac and Car's work [23] have performed verification test benefiting linear regression analysis and the graph of error scatter. In Fehlmann's work [24] Six Sigma methodology has been used on software production processes and the production cost has been reduced up to 75%. Performed works show that the selection of Six Sigma software affects the results of analysis [25], [26].

4. THE CONTROL SOFTWARE

4.1. Management of Acceptance Tests

Automated test control tool has been developed using Asp.Net. Fitnesse should be run on the side of server to manage acceptance tests. Using ProcessStartInfo object the jar file is executed with java -jar C:/Fit-NesseRoot/fitnesse-standalone.jar -p 8888 command. When this command is examined, we note that Fitnesse presents a port option. Fitnesse acceptance tests are held in FitnesseRoot folder with three types of file. The first called as "content.txt" keeps test information including library, class, function and values. Data which present the properties of test are kept in "properties.xml". Statistical information of performed tests are stored as zip files. The design of the system has been done with Asp.Net that facilitates the making transaction on these files.



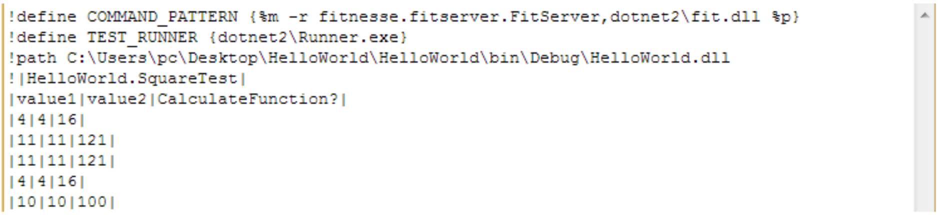
☑ **Assertions:** 4 right, 0 wrong, 0 ignored, 0 exceptions (0,002 seconds)

variable defined: COMMAND_PATTERN=%m -r fitnesse.fitserver.FitServer,dotnet2\fit.dll %p
variable defined: TEST_RUNNER=dotnet2\Runner.exe
classpath: C:\Users\pc\Desktop\HelloWorld\HelloWorld\bin\Debug\HelloWorld.dll

HelloWorld.SquareTest		
value1	value2	CalculateFunction?
4	4	16
11	11	121
11	11	121
11	11	121

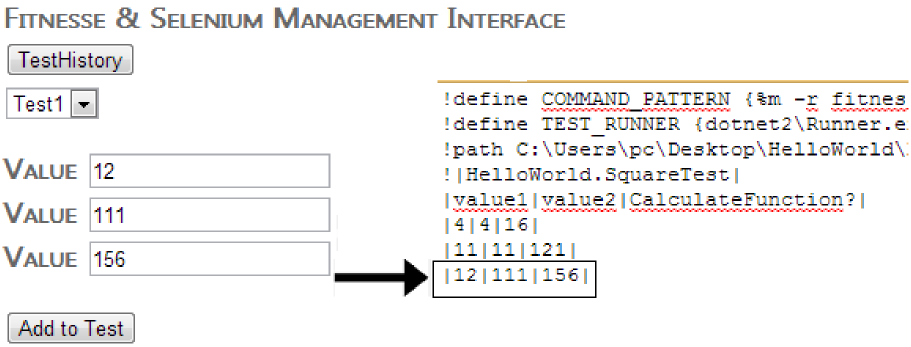
Figure 2. Main screen of automated test tool management.

As seen from Figure 2 desired test case is selected through DropDownlist control from the main screen of test management and test results are brought into the panel. Depending on the user preference test steps are proceeded over Fitnesse. The test arrangement table of Fitnesse is shown in Figure 3. Making directly operation on this table could increase the error rate rised because of writing mistakes. Also tests, which should be written in a certain order and format, require learning process. In developed framework, test data can be separately entered to system thus both test completion time and the error related to test writing format shorten as seen in Figure 4. Finished tests can be searched with it's name from test search page to delete from test case repository. Desired test to be deleted has been searched with name aftermath the properties and data of tests have been obtained as seen from Figure 5. These properties comprise test library, physical path, test function and value table.



```
!define COMMAND_PATTERN {%m -r fitnesse.fitserver.FitServer,dotnet2\fit.dll %p}
!define TEST_RUNNER {dotnet2\Runner.exe}
!path C:\Users\pc\Desktop\HelloWorld\HelloWorld\bin\Debug\HelloWorld.dll
!|HelloWorld.SquareTest|
|value1|value2|CalculateFunction?|
|4|4|16|
|11|11|121|
|11|11|121|
|4|4|16|
|10|10|100|
```

Figure 3. Fitnesse test arrangement table



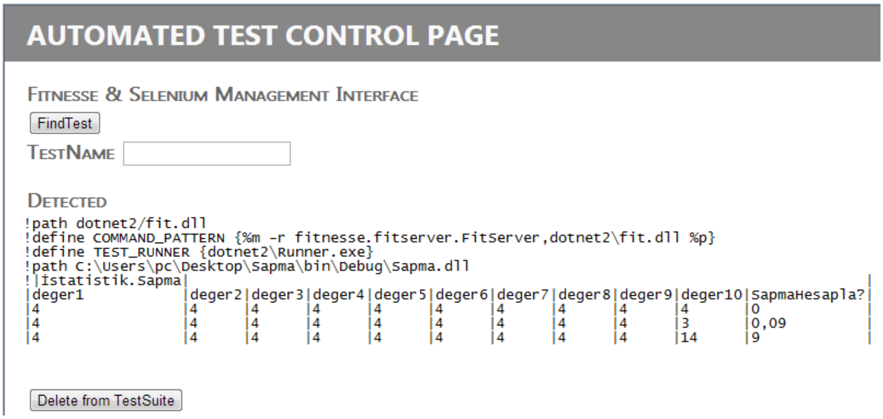
VALUE 12

VALUE 111

VALUE 156

```
!define COMMAND_PATTERN {%m -r fitnes
!define TEST_RUNNER {dotnet2\Runner.e
!path C:\Users\pc\Desktop\HelloWorld\
!|HelloWorld.SquareTest|
|value1|value2|CalculateFunction?|
|4|4|16|
|11|11|121|
|12|111|156|
```

Figure 4. Test addition page


AUTOMATED TEST CONTROL PAGE
FITNESSE & SELENIUM MANAGEMENT INTERFACE
FindTest
TESTNAME
DETECTED
Delete from TestSuite

```
!path dotnet2/fit.dll
!define COMMAND_PATTERN {%m -r fitnesse.fitserver.FitServer,dotnet2\fit.dll %p}
!define TEST_RUNNER {dotnet2\Runner.exe}
!path C:\Users\pc\Desktop\Sapma\bin\Debug\Sapma.dll
!|İstatistik.Sapma|                                                                   |
|deger1               |deger2|deger3|deger4|deger5|deger6|deger7|deger8|deger9|deger10|SapmaHesapla?|
|4                    |4     |4     |4     |4     |4     |4     |4     |4     |4      |0            |
|4                    |4     |4     |4     |4     |4     |4     |4     |4     |3      |0,09         |
|4                    |4     |4     |4     |4     |4     |4     |4     |4     |14     |9            |
```

Figure 5. Page of deleting test

Through developed framework also similar test data can be scanned which available in test files thus test data analysis can be accomplished as seen from Figure 6. This analysis provides the determination of test functions which have similar test data. While the analysis is being performed, the largest and the smallest test data of test groups have been randomly searched. This method expands the bounds of tests thus the results of analysis are becoming more consistent. In completed analysis, five different Fitnesse test data have been searched in five different Fitnesse test group in terms of percent. Test group consists of DataSet1, DataSet2, DataSet3, DataSet4 and DataSet5. Each of these test groups have 40 different test data. Randomly selected 10 data have been calculated pulling from every data groups. Experimented data include TData1, TData2, TData3, TData4 and TData5. According to analysis result, despite TData2 and TData3 are in different test groups, the distribution of these data seem to be similar. Benefiting this similar distribution a conclusion comes out that TData2 and TData3 can be merged. Merged test groups help to reduce effort which heavily allocated for test. Also this merger facilitates test tracing control while test harness. Variance table of test data sets is seen in Chart 2. As illustrated in this table, the variance values of TData2 and TData3 are close to each other that depicts the evaluation should be done in the same test data group.

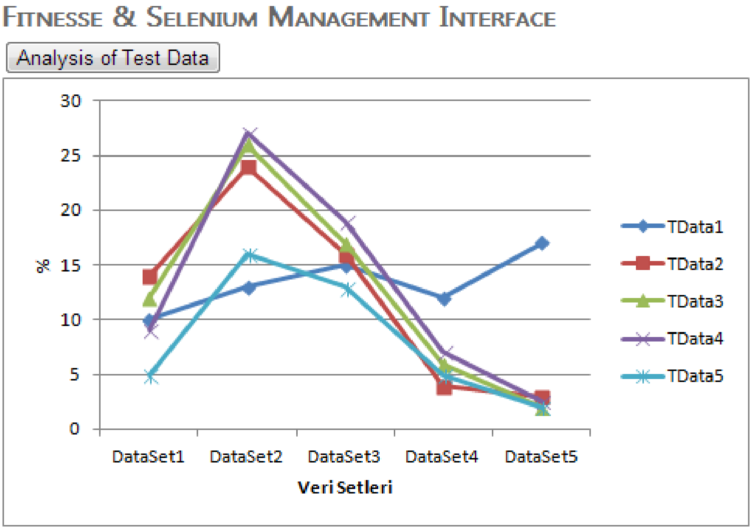


Figure 6. Analysis of test data.

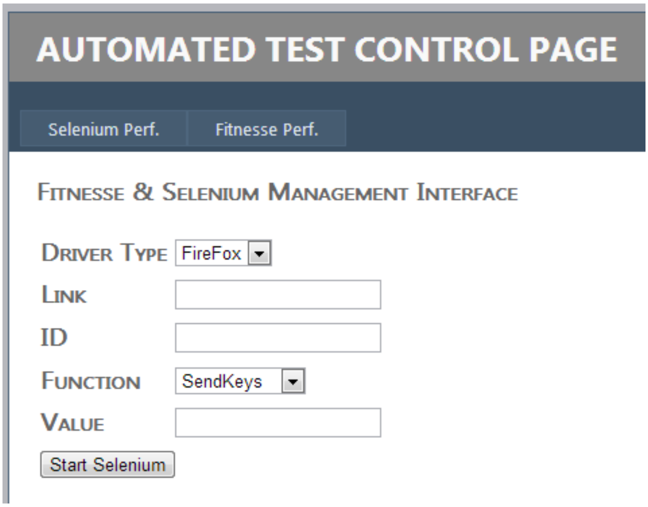
Chart 2. Variance table of test data

Data Set	σ^2
TData1	3.31
TData2	1.64
TData3	1.62
TData4	2.45
TData5	2.79

The analysis results would be well understood if the test data had been augmented. To generate automated acceptance tests MATLAB is also becoming preferable [20]. MATLAB is to be known as effective for numerical data operation. Because of the acceptance is performed through the comparison of actual and expected value, a novel test framework can be designed via a simple matrix table and MATLAB functions.

4.2. Management of GUI Tests

The usage of OpenQA.Selenium libraries is needed to manage automated GUI tests. After the selection of browser which to be implemented for GUI test, test operation is accomplished with the entering of some information such as url, id, function. As seen from Figure 7 one of the browser including IE, FireFox and Chrome can be selected for webdriver. A function could be executed such as SendKeys, Click, Clear, GetAttribute. If the function requires an input, after the entering of related input the test is run. In Selenium IDE plugin, test transactions can be conducted as recorded test. The most obvious benefit of this test specification is the execution of GUI tests by user who has general information about page structure without any test harness information.


AUTOMATED TEST CONTROL PAGE

Selenium Perf. Fitnesse Perf.

FITNESSE & SELENIUM MANAGEMENT INTERFACE

DRIVER TYPE: FireFox

LINK:

ID:

FUNCTION: SendKeys

VALUE:

Start Selenium

Figure 7. Test specification page

The processes which occur during the Selenium GUI test management can be expressed as follows:

- Inclusion of the library of Selenium OpenQA
- Generation of WebDriver profile
- Entering desired url to the WebDriver object
- Finding desired web element thanks to the WebDriver
- Execution of stored web element which found through ID information

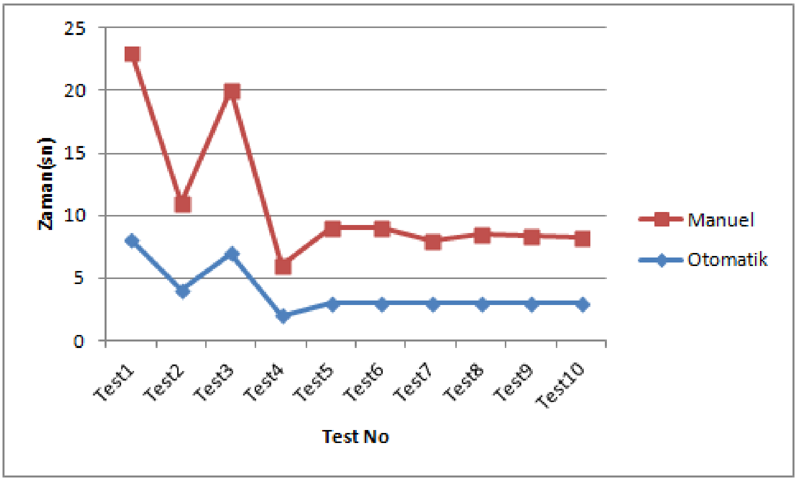


Figure 8. Time graph of test management

4.3. Statistical Analysis

4.3.1. Variance Analysis (Anova)

Variance analysis called as ANOVA firstly developed and introduced by R.A. Fisher who is one of the British statistician. Initially this method was applied to agricultural data [29], later on large areas for data analysis. To measure data fluctuation on every level, ANOVA can be used for the calculation of test data which intended to different size and goal. The main goal of this analysis is to detect the modifications of mean value on every data level. In this regard, four type of ANOVA techniques are available:

- One-way among the groups,
- One-way repeated measurements,
- Two-way among the groups,
- Two-way repeated measurements.

One-way analysis, which available in these techniques, has one independent categorical variable and one continuous variable. Independent variable consists of a few group levels. The analysis has two steps if independent variables greater than two levels. Firstly F test is applied to determine difference among the means. If F value is important in accordance to statistic, the difference are tried to be found among two means. If F value is under 0.05 we can say that groups are different with respect to statistical evaluation. In one-way repeated measurement, a group of experiment data is investigated under the three or more experimental situation. While the experimental situation is performed, the variation of means is examined over a specific time. In two-way analysis the classification is conducted using two categorical variables. Test data has been classified using two-way analysis seen in Chart 3. Two-way repeated analysis is performed based on two-way analysis with respect to time in various experimental situations.

Chart 3. Two-way analysis table

	Similarity ratio %10-20	**Similarity ratio %20-25**	**Similarity ratio %25-30**
Training data	TData5, TData1	TData3, TData2	TData4
Test data	TData1	TData1	TData1
	TData2	TData2	TData2
	TData3	TData3	TData3
	TData4	TData4	TData4
	TData5	TData5	TData5

Before the analysis, we should detect whether the data is categorical or continuous. Non-parametric tests are applied to categorical data, on the other hand parametric tests are applied to continuous data. The data shown in Figure 5 express data sets including 10 data set. ANOVA has been selected for analysis because used data set is suitable for parametric analysis.

4.3.2. Xbar-R Graph

To analysis test data XBAR/R and ANOVA have been benefited. Generally a process can be tracked thanks to control charts [27] such as X-R, P. Sudden changes which occur tracked processes and interruptions can be monitored by these graphs. Such graphs firstly tried by Dr. Walter A. Shewart at Bell laboratory [28]. These graphs are effective for tracking whether sharp changes are available. XBar-R has been selected because the data size consists of 10 data set. Selected X-R graph used for tracking automated test data is seen in Figure 9. Second graph of Figure 9 illustrates control limits on the other hand first graph illustrates the distribution of test data. The control limits are not exceeded in changed data. We conclude from this that test process is stable. In Chart 4 multi comparison is seen. Three sigma has been used because the trust value of results is obtained at third level. In this analysis similar test data generate one group and dissimilar test data generate the other group. According to the results obtained from comparison analysis, because of difference value is 0.002 that less than 0.05 two test data groups are different from each other.

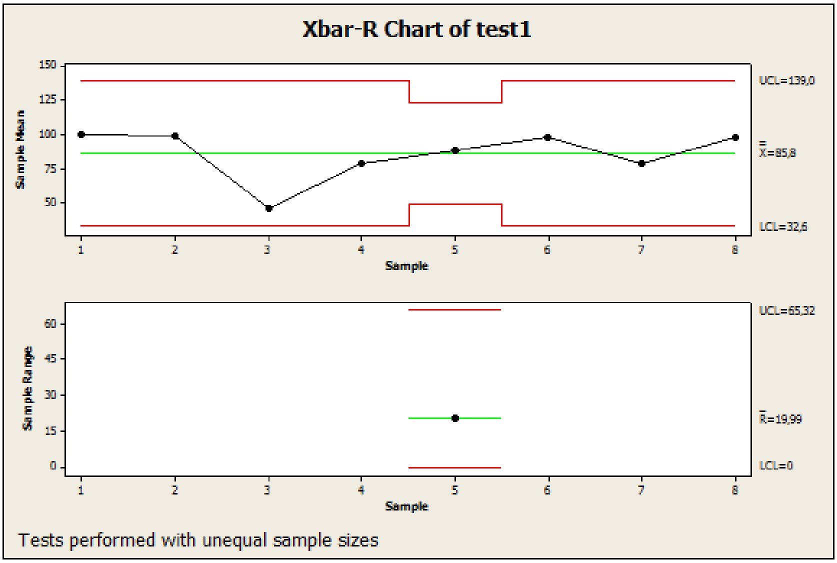


Figure 9. X-R graph of test data

Figure 10 shows the results of ANOVA including TDATA1, TDATA2, TDATA3, TDATA4, TDATA5 which selected from test data sets. While assigning these values to test data vary between 0-80000 some properties have been considered such as test data size, count of functions, class count and usage count. 60000 is a threshold value that determines the normality of data. Test data should not fall below the limit value determined in our analysis as 20000. TDATA1, TDATA2 and TDATA3 test data don't exceed the limit value (60000) that we can say the ANOVA analysis is normal. Showing the homogeneous distribution of the test data expresses the test data suitable for normal distribution. TDATA4 exceeds the limit value thus this data don't obey normality. Hence this test data should not be selected to obtain healthy test results. TDATA5 has come to limit value expressed as 20000. But this data can be used for test data because the stable value has not fallen below this value. The detailed comparison of test results can be found in Chart 4 including sum of squares, df, mean square, f and significance.

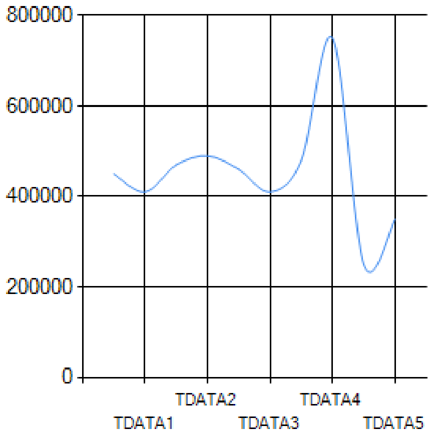


Figure 10. Variance analysis of test data

Chart 4. ANOVA multi comparative test result

Sum of squares	df	Mean square	F	Significance
Between groups 2.950	3	0.924	5.042	0.002
Within 3.670 groups	15	2.42		
Total 6.620	18			

5. CONCLUSION

Usage of automated tests is one of the way to improve software quality. With an automated test tool, it is possible that automated tests can be easily managed and designed. The framework presented in this paper aims to complete specified needs. The main advantage of designed framework is to manage acceptance tests with GUI tests. Development of framework using .Net has facilitated the management of two different test tool. The effort allocated for test addition and deleting has been reduced thanks to test data saving options including xml, xhtml and .txt. Ten test scenarios have run on both software testing tools and proposed framework. Selected scenarios are distinctive with respect to functionality and test goal. Measurement performed using testing tools is called as “Manual”, in the other hand measurement performed using proposed framework is called as “Automated”. To obtain an accurate performance evaluation C# StopWatch object has been used, elapsed times which obtained using testing tools have been saved. Aftermath the time comparison has been done based on second among saved times and new times obtained using proposed framework. As seen from Figure 8 automated test management tool has reduced operational period up to two times when it is compared with traditional usage. ANOVA and control charts can be used for the selection of test data. To extends this work other statistical methods should be tried and compared with old results. Designed system is based on web that could shed new light mobile test management system to be designed. The effectiveness of proposed automated test tool can be increased using parallel programming techniques. In this respect, TPL presented in .Net 4.0 will help to extension of this paper.

REFERENCES / KAYNAKLAR

[1] Besson, F. M., Beder, D. M., Chaim, M. L.: An Automated Approach for Acceptance Web Test Case Modeling and Executing, Agile Processes in Software Engineering and Extreme Programming, 48, 160-165, 2010.

[2] Larsen, M.:Get the Balance Right: Acceptance Test Driven Development, GUI Automation and Exploratory Testing, PNSQC 2012, 1-14, 2012.

[3] Adzic, G.: Test Driven. NET Development with FitNesse, Lulu Enterprises Uk Limited, 2008.

[4] Shahamiri, S., Kadira, W., Ibrahima, S., Hashim, S.: An automated framework for software test oracle. Information and Software Technology, 53, 2011.

[5] Pocatilu, P.:Automated Software Testing Process, Economy Informatics, 2(1), 97-99, 2002.

[6] Kim, E. H., Na, J. C., and Ryoo S. M.: Implementing an Effective Test Automation Framework, Computer Software and Applications Conference, 2, 534-538, 2009.

[7] Badle, S., Selenium Web Browser Automation Site, Available from: http://docs.seleniumhq.org/, 2013.

[8] Wang, Xinchun, and Peijie Xu: Build an auto testing framework based on selenium and fitnesse., ITCS International Conference onInformation Technology and Computer Science, 2, 2009.

[9] Richardson, A.:Selenium Simplified, A tutorial guide to using the Selenium API in Java with JUnit, London, 2010.

[10] Ozturk, M. Maruf: Distance education measurement and evaluation system design and performance analyze with software testing techniques, published master dissertation, 2012.

[11] Bruns, A. Kornstadt, A., Wichmann, D.: Web Application Tests with Selenium, IEEE Software, 88-91, 2009.

[12] Catelani, M., Ciani, L., Scarano, V. L., Bacioccola, A.: Software automated testing: A solution to maximize the test plan coverage and to increase software reliability and quality in use, Computer Standards & Interfaces, 33(2), 152-158, 2011.

[13] Huttermann, M.:Quality and Testing, DevOps for Developers, 51-64, 2012.

[14] Pan, Z., Park, H., Baik, J., Choi, H.: A Six Sigma Framework for Software Process Improvements and its Implementation. In: Software Engineering Conference. IEEE, Los Alamitos, 2007.

[15] Löffler, R., Gülldali, B., Geisen, S.:Towards Model-based Acceptance Testing for Scrum. Softwaretechnik-Trends, GI, 2010.

[16] Collins, E., Dias-Neto, A.,de Lucena, V.F.:Strategies for Agile Software Testing Automation: An Industrial Experience, 440-445, 2012.

[17] Dale H. Emery. Writing Maintainable Automated Acceptance Tests. http://dhemery.com/pdf/writing maintainable automated acceptance tests.pdf, 2013. [Online; accessed 14-June-2013].

[18] Xie, T.:Improving Effectiveness of Automated Software Testing in the Absence of Specifications, In Proceedings of the 22nd IEEE international Conference on Software Maintenance, 355-359, 2006.

[19] Jureczko, M., Mynarski, M.: Automated acceptance testing tools for web applications using test-driven development. Electrotechnical Review, 86(9), 198-202, 2010.

[20] S.L. Eddins: Automated Software Testing for Matlab. IEEE Computing in Science, 2009.

[21] Head, G.E. "Six-Sigma Software Using Cleanroom Software Engineering Techniques." Hewlett-Packard Journal 1994 (June 1994): 40–50.

[22] Cvetan Redzic and Jongmoon Baik, “Six Sigma Approach in Software Quality Improvement”, International Conference on Software Engineering Research, Management and Applications (SERA'06), August 2006.

[23] T. Galinac, Ž. Car, Software verification process improvement proposal using Six Sigma Proceedings of 8th International Product-Focused Software Process Improvement Conference, Lecture Notes in Computer Science, 4589, 51–64, Springer-Verlag, Berlin, 2007.

[24] Fehlmann, Th. (2004), “Six sigma for software”, Proceedings of the 1st SMEF Conference, Rome.

[25] Arul, K., Kohli, H., “Six Sigma for software application of hypothesis tests to software data”, Software Quality Journal, 12(1), 29-42, 2004.

[26] D. E. Harter, C. F. Kemerer, and Sandra A. Slaughter, “Does software process improvement reduce the severity of defects? A longitudinal field study”, IEEE Trans. Software Eng., 38(4), 810-827, 2012.

[27] Champ, C.W. and Woodall, W.H., Exact Results for Shewhart Control Charts with Supplementary Runs Rules, Technometrics, 29, 1987.

[28] Shewhart, W.A., Economic Control of Quality of Manufactured Product, Van Nostrand-Reinhold, NY, 1931.

[29] Fisher, R.A., Statistical Methods for Research Workers. Oliver & Boyd, Edinburgh, 1925.

[30] Glidden, David V., Stephen C. Shiboski, and Charles E. McCulloch. Regression methods in biostatistics: linear, logistic, survival, and repeated measures models. Springer, 2011.