Revista Cubana de Ciencias Informáticas
Vol. 16, No. 2, Abril-Junio, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 101-117

Artículo original

Good practices for software test automation. Case study in an electronic store

Buenas prácticas para la automatización de pruebas de software. Caso estudio en una tienda electrónica

Sandra Verona Marcos [1*] https://orcid.org/ 0000-0002-2794-092X
Martha Dunia Delgado Dapena [2] https://orcid.org/ 0000-0002-2601-3462
Alejandro Miguel Güemes Esperón [3] https://orcid.org/ 0000-0001-9704-9449
Melissa Castillo Mendoza [4] https://orcid.org/ 0000-0003-4756-9176

[1] Universidad Tecnológica de La Habana "José Antonio Echeverría", CUJAE. Calle 114 #11901 /Ciclovía y Rotonda CP 19390. sverona@ceis.cujae.edu.cu
[2] Universidad Tecnológica de La Habana "José Antonio Echeverría", CUJAE. Calle 114 #11901 /Ciclovía y Rotonda CP 19390. marta@ceis.cujae.edu.cu
[3] Universidad Tecnológica de La Habana "José Antonio Echeverría", CUJAE. Calle 114 #11901 /Ciclovía y Rotonda CP 19390. aguemes@tesla.cujae.edu.cu
[4] Universidad Tecnológica de La Habana "José Antonio Echeverría", CUJAE. Calle 114 # 11901 /Ciclovía y Rotonda CP 19390. mcastillo@ceis.cujae.edu.cu

*Autor para la correspondencia.(sverona@ceis.cujae.edu.cu)

ABSTRACT

Testing represents a fundamental activity in the software development life cycle and, in many cases, it is practically the only means used in projects for software verification and validation. Despite this, traditionally, this stage has been carried out at the end of the process, when the product is finished and is about to be released; which brings with it delays in deliveries and customer dissatisfaction among other negative aspects. In the present work, a case study is used to show the obtaining of Test Cases from User Stories, using black box techniques. A study of the different test automation tools is also carried out and the use of them in the implementation of the designed tests is shown.

Keywords: software quality; software testing; reduced test suites.

RESUMEN

Las pruebas representan una actividad fundamental en el ciclo de vida del desarrollo de software y, en muchos casos, suponen prácticamente el único medio empleado en los proyectos para la verificación y validación del software. A pesar de ello, tradicionalmente, esta etapa se ha llevado a cabo al final del proceso, cuando el producto está terminado y está a punto de ser liberado; lo cual trae consigo demoras en las entregas e insatisfacciones en los clientes entre otros aspectos negativos. En el presente trabajo se utiliza un caso de estudio para mostrar la obtención de los Casos de prueba partir de Historias de Usuario, utilizando técnicas de caja negra. Se realiza además un estudio de las distintas herramientas de automatización de prueba y se muestra la utilización de ellas en la implementación de las pruebas diseñadas.

Palabras clave: calidad de software; pruebas de software; suites de pruebas reducidas.

Recibido: 02/02/2022

Aceptado: 22/03/2022

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 2, Abril-Junio, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 101-117

Introduction

Testing represents a fundamental activity in software development and, in many cases, it is practically the only means used in projects for software verification and validation. Myers in (Myers, 2011) defines software testing as “a process, or series of processes, designed to ensure that the source code does what it is designed to do and therefore does not do anything unexpected”.

The main objective of software testing is to run a program with the intention of detecting errors. According to Pressman, the discovery of the largest number of errors in a software product requires developing tests in a systematic way and designing Test Cases (TC) using defined techniques. It also states that the development of tests is considered one of the most expensive processes in the software life cycle, between 30% and 40% of the total effort, due to the weight they have in certifying the quality of the developed products (Pressman, 2015).

Testing software is one of the most important activities in the development life cycle, but it has traditionally been carried out at the end of the process, when the product is finished and about to be released. The complexity of today's software requires testing to run parallel to development, so that bugs are found early and can be fixed at low cost (Piattini, 2010). The quality of the software must be taken into account throughout its production process, therefore, there must be activities that ensure compliance throughout its life cycle. It is a bad practice to wait for the software to be finished and then aspire to obtain quality in it (Sommerville, 2011) (Jústiz, 2014).

Software bugs can have drastic consequences if they are not dealt with in time. There are many examples of this and in all these situations a good analysis at the right time could have prevented the failures. Although there are tests that must be carried out manually, there are numerous tools, both proprietary and open source, that assist the testing team.

The good use of automation tools, having a structured process during the development of functional tests and efficient manual tests, are the ideal complement to carry out a greater coverage of tests in the different systems, thus ensuring the quality of the software, In addition to providing benefits to companies such as; structure the tests, greater detection of defects, more

efficient feedback to the development team, early stabilization of the source code, since by finding errors faster the development team has time to stabilize it and finally the results they produce are reliable.

Test automation is an effective tool to reduce the time and costs of a test plan in projects that generate software products. Since the first automation test tools appeared, they have been the center of attention for the advantages they offer in reducing of human errors during the testing stage of the development cycle, in addition to allowing people through this process to acquire new technical and professional skills, in turn, companies acquire the challenge of implementing this process in their companies efficiently achieving better quality of software products (Łukasz, 2018) (Colorado, 2020).

In this work, good practices are proposed for the design of a test suite from user stories (US) to its execution in a tool, shown through a case study of an electronic store.

Related Jobs

Techniques such as test-driven development (TDD) and behavior-driven development (BDD) are fundamental pieces that guarantee quality and integrity during the life cycle of a product (Wynne, 2012). TDD is an iterative object-oriented software design practice, which was introduced by Kent Beck and Ward Cunningham as part of Extreme Programming (XP). Basically, it includes three subpractices (Fontela, 2011):

Automation: the tests of the program must be done in code, and with the single run of the test code we must know if what we are testing works well or badly.

Test-First: Tests are written before the code to be tested.

Post refactoring: to maintain the quality of the design, the design is changed without changing the functionality, keeping the tests.

In TDD, developers write tests to specify what a unit of code should do, then implement this unit of code to satisfy the requirements. These tests are called unit tests, and they focus on small units of code.

BDD is a development practice that stems from agile development methodologies. At its core it is a refinement of TDD. Some features of BDD are (de Florinier, 2011):

You're not just defining tests; you're also defining behaviors. Improve communication between testers, users and developers. Because BDD uses a simplified and common language, the learning curve is shorter than TDD.

The search-based Automatic Test Generation model, MTest.search, is an environment for executing software tests at different phases of software product development, which is independent of the development approach or methodology used in the process. productive. This environment is made up of integrating Workflows for the execution of early tests in the production environment, the central core of TC generation for obtaining test code, and the Integrated Automated Tools that allow support for the execution of the flows. of work (Delgado, 2017). This proposal provides a flow for the design of the TC starting from the specification of requirements. This flow can be taken into account for the design, but it is necessary to adapt it to apply it in the case of obtaining TC from US.

Test automation tools

The automation of tests emerged as an alternative to speed up their execution, as well as to improve the reliability of the product and its quality (Polo, 2015). There are different test automation tools: JMetter, Selenium IDE, Selenium Web Drive, SoapUI, Katalon Studio, among others. JMetter is a tool dedicated to creating, executing and monitoring the course of load, overload and performance tests, as well as functional tests (Łukasz, 2018). You can test static and dynamic resources including SOAP/REST web services, HTTP and HTTPS websites, databases, FTP, and mail servers. It works by simulating the load on the server to analyze the overall performance of the application/website under test. With very little technical knowledge, the

results provided by this tool can be interpreted (Tzancoff, 2018). Among the advantages and disadvantages of JMetter are (Łukasz, 2018):

Advantages, Ability to test many types of systems, as well as high scalability to increase the load range.

Disadvantages, Realistic client/server configuration mapping required, test machine limitations may appear in latency in response times, tool is resource intensive.

Katalon Studio is a free, automated testing tool that can be installed on MacOS and Windows based on Selenium and Appium testing frameworks (Ranadhan, 2020). The tool does not require any programming knowledge. However, in the case of testing mobile applications, it is good to know mobile technologies such as Android, and in the case of network services, methods of communicating with the server. The tool allows TC to be bundled and stored and interpreted separately, so it can be used for the implementation of professional automated tests (Łukasz, 2018). Among the functionalities offered by this tool are (Ranadhan, 2020):

Spy object, capture objects for mobile and web based applications. Data-driven and test data management, provide object data files that come from csv, excel, relational db files. Recording and generation of tests, recording and creation of test script based on what happened during the recording process. Reports and Analysis, Katalon Studio report results can be exported to csv, html and pdf files. Among the advantages and disadvantages of Katalon Studio are (Łukasz, 2018):

Advantages, Ability to test mobile, web and web services applications in multiple browsers.

Disadvantages, Unpopular tool, additional paid technical support.

Selenium IDE is a free and open source plugin for the Firefox web browser, which can be easily downloaded using the Selenium website. It was initially used by the web development community to automatically test web applications (Holmes, 2006). Selenium IDE is a web testing tool that uses scripts to run tests directly in the browser. The tool uses JavaScript and iFrames to embed automatic testing mechanisms in the browser, this allows the same test script to be used to test multiple browsers on multiple platforms. It is a very reliable tool, it offers user support through its websites, with documentation accessible to all (Kaur, 2013). It is one of the most widely used

frameworks for testing web applications, mainly for web interface and functional testing. It is very easy to use since it is not necessary to learn the test programming language (Chopra, 2012).

It is important to choose a testing tool that allows you to fully complete the automated testing process. As a result of this, a group of indicators was identified to make a comparison between the three selected tools.

- Object Spy: Allows you to manually capture the interface elements you want to get and store them in the page elements in the object library.
- Test recording: allows the recording of the test script.
- Keywords: keyword management to cover the operations used in the tests.
- Generate reports: offers generation of reports and reports with the results of the tests executed.
- Documentation and help: forums, web pages and community that offer information about the tools.
- Data File Object: Allows you to get data from external sources. Data sources can support multiple formats: Excel, csv, databases and others.
- Test script generation: allows you to generate the test script that can then be modified.

Table I shows the comparison made between the three test execution tools studied, where it is observed that Katalon Studio and Selenium are compatible with the web spy object that allows registering objects in web applications. All three tools have data file objects to query data from external sources: databases, Excel, CSV. In the case of Selenium IDE you need to download a plugin to use external data sources or you need to code the test data directly. These tools provide test recording and scripting to save time running tests. JMetter has no keywords while Katalon Studio and Selenium have a set of keywords that allow most test cases to be implemented. The Katalon Studio tool, in addition to offering keywords, allows you to create custom words to use them in the same way as those offered by the tool. Regarding the generation of reports, Katalon offers readable and easy to understand reports that can be exported to HTML, PDF and CVS files. For its part, JMeter also generates reports in graphs and detailed reports, while Selenium only provides simple report templates. These tools have a large community and documentation

that can be found on their official websites. Performance wise Katalon and JMetter can be a bit slower compared to Selenium. The JMeter tool does not allow you to program tests before coding, which goes against test-driven software development, a fundamental approach to follow for software development with agile methodologies.

The tools by themselves do not solve the problem of detecting errors or defects, to achieve high effectiveness it is necessary that the TCs that are implemented in the execution tools have input values that allow these errors to be found. Table 1

Table 1 - Comparison of the automatic test execution tools studied.

Indicadores	Katalon Studio	Selenium IDE	JMeter
spy item	X	X	
test recording	X	X	X
Test script generation	X	X	X
Keywords	X	X	
Generate reports	X	X	X
Documentation and help	X	X	X
data file object	X	Need to download a plugin	X

Test Case Design Techniques

TC design techniques are directed in this direction. Functional tests or also known as black box tests, do not evaluate the internal behavior of the system nor do they have access to the source code, their objective is to evaluate the external behavior of the developed program, analyzing and verifying the results according to the functional specifications and customer requirements (Pressman, 2015).

Black box testing techniques are described below (Pressman, 2015):

1. **Equivalence Partition:** Performs TC from a set of software inputs, which are converted into valid and invalid data classes for each input condition. Some authors describe input value conditions as: numeric values, range of values, set of related values, boolean value.

2. **Boundary Value Analysis:** A large number of errors are usually found at the limits of a value, which, in the center, is why boundary value analysis was developed as a TC design technique that complements equivalence partitioning. TCs are designed by taking values on the border of the input and output values of the equivalence class. The guidelines for boundary value analysis are similar to those for equivalence partitioning, the difference is that the values **a** and **b** assigned to the specified input condition of a range or specific inputs of number of values, are above and below the assigned values of **a** and **b**.

3. **Cause and Effect Graph:** Helps the systematic choice of TC. Defining as Cause (conditions of inputs or actions of 30 users) and effect (expected actions of the system). The Cause and Effect graphs are based on the specifications, identifying the cause and effect, transforming them into Boolean graphs, converting them into a decision table and Converting the rules to TC.

Materials and methods

Among the good practices proposed in this work are those of combining the flow proposed in (Delgado, 2017) with test case execution tools. The flow was modified as explained below.

Figure 1 shows the process that was followed to obtain the TC by modifying the diagram of the Mtest.search Model.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 2, Abril-Junio, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 101-117

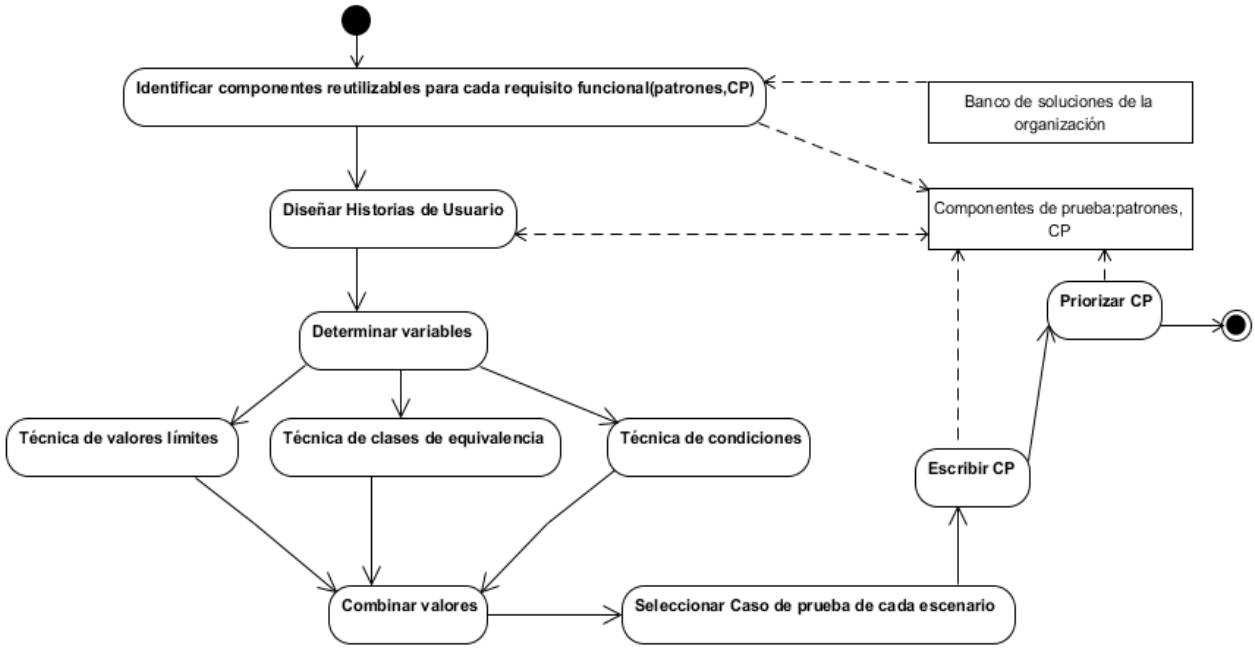

Fig. 1 – Process for obtaining Test Cases.

From each element of the US template, the necessary information for the design of the TC is obtained. The conditions described in the scenarios and the variables defined in the US are used to apply the techniques that provide the combinations of values necessary to write the TC. From the domain of each variable, the techniques of partition equivalences and limit values are applied to obtain a set of significant values. These values are later extended by applying the conditions technique to the conditions that appear in the description of each US scenario. Based on this information, the TCs are written in the execution tool, depending on the type of tool, and then the test suite is executed.

To show the application of these good practices, a case study was designed that consists of applying the entire procedure to an electronic store application and evaluating how to execute the tests of this application in three different execution tools.

Case Study: Electronic Store

Questions:

How to apply the good TC design practices proposed to obtain the TC from the US?

How are designed tests implemented in an execution tool?

The following requirements were selected to describe the US together with the clients:

1. Authenticate User.
2. Register User.
3. Product display.
4. Return of products and/or services.
5. Form combo.
6. Activate products.
7. Register user in the administration panel.
8. Make purchase.
9. Synchronization of products.
10. Configure transport.
11. Configure payment method.
12. Configure delivery method.
13. List orders for a user.
14. List commands for admin panel.
15. Configure store.
16. Manage traces.
17. Product Report of its minimum and maximum quantities (stock limit)

Results and Discussion

Based on the users' proposals, a first design of the USs was made, they were reviewed and refined in several iterations until the final description of 16 USs was obtained. To obtain the US, five clients were chosen.

The programming tasks were obtained from the USs and approved by the development team. To generate the 39 programming tasks from the US, a task was generated for each different scenario of the US.

To obtain the TC, the techniques of equivalent partitions, limit value and the technique of conditions were used with the help of the logical conditions obtained from the collected US.

The applied techniques and the TC obtained from one of the US are detailed below as an example.

Table 2 shows one of the USs captured together with the client:

Table 2 - US1 Authenticate User.

Identifier	**Name US**	**Priority**
US1	Authenticate User	1
User: User		
Description: The user needs to be able to authenticate to access the system.		
Responsible programmer:		
Scenarios		
Name	**Condition**	**Expected result**
Successful user authentication	If Username exists and Password is correct and Captcha is correct.	You access the system having the visibility allowed by the role you occupy.

Revista Cubana de Ciencias Informáticas
Vol. 16, No. 2, Abril-Junio, 2022
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 101-117

User authentication failed due to incorrect credentials	If Username does not exist and/or Password is incorrect and/or Captcha is incorrect and QtyAttempts > 3.	A message is displayed indicating that the user has been blocked because they exceeded the maximum number of attempts.
User authentication failed due to incorrect credentials.	If Username does not exist and/or Password is incorrect and/or Captcha is incorrect and QtyAttempts <3.	A message is displayed indicating that the credentials are incorrect to try again.
User authentication failed due to null credentials.	If any of the data: Username, Password and Captcha is null.	A message is displayed indicating that it is necessary to fill in the corresponding data to authenticate in the system.
Variables		
Name	**Type or Domain**	**Description**
Username	String that supports letters, symbols, and numbers.	Capture the name of a user.
Password	String that supports letters, symbols, and numbers.	Capture a user's password.
Captcha	String that supports letters, symbols, and numbers.	Capture the sequence (Captcha) to verify that the user is a human.
CantIntents	Numeric, does not support decimals.	Captures the number of attempts the user has to access the system

From each element of the US template, data was obtained for the design of the PCs. The conditions described, the scenarios and the variables defined in the US were used to apply the techniques that provided the combinations of values necessary to write the TC. From the domain of each variable, the technique of equivalence partitions and limit values was applied to obtain a set of significant values. These values were later extended by applying the conditions technique to the conditions that appear in the description of each US scenario. For the US shown in Table 2, the significant values shown in Table 3 and Table 4 were generated.

Table 3 - Limit value technique for the Authenticate User US.

Limit value			
Value	**Condition**	**Variable**	**value**

3	<	CantIntents	1
3	=	CantIntents	3
3	>	CantIntents	4

Table 4- Technical conditions for US Authenticate User.

Conditions technique			
Terms	**Type**	**Condition**	**Variable**
Username must be correct	Logical condition	True/false	Username
Username must not be null	Logical condition	True/false	Username
The password must be correct	Logical condition	True/false	Password
Password cannot be null	Logical condition	True/false	Password
The captcha cannot be null	Logical condition	True/false	Captcha
The captcha must be correct	Logical condition	True/false	Captcha
The number of attempts to access the system must be <3	Range	True/false	CantIntents

Table 5 shows the TC generated from the combinations of values obtained with the different techniques.

Table 5 - TC for US Authenticate User.

Combinations of input values				**Expected results**	**Real Results**
TC	stage	input variable name	value		
1	Successful user authentication	Username	jrperezz	You access the system having the visibility allowed by the role you occupy.	
		Password	jrperez*10		
		Captcha	TcdnXm		
		CantIntents	1		

2	User authentication failed due to incorrect credentials	Username	jrperezz	A message is displayed indicating that the user has been blocked because they reached the limit of allowed attempts	
		Password	jrperez10		
		Captcha	TcdnXm		
		CantIntents	3		
3	User authentication failed due to null credentials	Username		A message is displayed indicating that it is necessary to fill in the corresponding data to authenticate in the system.	
		Password	jrperez*10		
		Captcha	TcdnXm		
		CantIntents	1		
4	User authentication failed due to incorrect credentials.	Username	jrperezz	A message is displayed indicating that the credentials are incorrect.	
		Password	jrperez10		
		Captcha	TcdnXm		
		CantIntents	1		
5	User authentication failed due to null credentials.	Username	jrperezz	A message is displayed indicating that it is necessary to fill in the corresponding data to authenticate in the system.	
		Password			
		Captcha	TcdnXm		
		CantIntents	1		
6	User authentication failed due to incorrect credentials.	Username	jrperezz	A message is displayed indicating that the captcha code is incorrect.	
		Password	jrperez*10		
		Captcha	tcfnaZ		
		CantIntents	1		
7	User authentication failed due to null credentials.	Username	jrperez	A message is displayed indicating that it is necessary to fill in the corresponding data to authenticate in the system.	
		Password	jrperez*10		
		Captcha			

		CantIntents	1		
8	User authentication failed due to null credentials.	Username		A message is displayed indicating that it is necessary to fill in the corresponding data to authenticate in the system.	
		Password			
		Captcha			
		CantIntents	2		
9	User authentication failed due to incorrect credentials.	Username	rperez	A message is displayed indicating that the credentials are incorrect.	
		Password	jrperez10		
		Captcha	TcdnXm		
		CantIntents	4		
10	User authentication failed due to incorrect credentials.	Username	rperezz	A message is displayed indicating that the credentials are incorrect.	
		Password	jrperez*10		
		Captcha	tcfnaZ		
		CantIntents	1		
11	User authentication failed due to incorrect credentials	Username	jrperezz	A message is displayed indicating that the credentials are incorrect.	
		Password	jrperez1		
		Captcha	tcfnaZ		
		CantIntents	1		

Conclusions

With the development of the work, the technologies associated with the automatic execution of software tests for web environments were assimilated, which allows selecting the most appropriate depending on the productive environment. This helps streamline the testing process in the software product development cycle. Test case design techniques were applied from User Stories captured with the client, to obtain the Test Cases of the case study used; which allowed taking into account values that help detect a greater number of errors. Also, the test scripts were implemented in the automated test execution environments.

References

Myers, G. J. T. B.; Et Al. The Art Of Software Testing. New Jersey: Johnwiley & Sons, 2011. 978-1-118-03196-4.

Pressman, R. S. Ingeniería Del Software. Un Enfoque Práctico. Mexico Mcgraw Hill: Mcgraw Hill, 2015. 978-607-15-0314-5.

Piattini, M. G. G., F. O.; Guzmán I. G. & Pino F. J. Calidad De Sistemas De Información. 2da Edición Actualizada. Ra-Ma Editorial, 2010. 978-84-9964-070-9.

Sommerville, I. Software Engineering. Boston, Massachusetts: Addison-Wesley, 2011. 978-0-13-703515-1.

Jústiz-Núñez, Delgado-Dapena, M. D.: Proceso De Pruebas Para Productos De Software En Un Laboratorio De Calidad, Ingeniería Industrial, Vol. Xxxv, Pp. 131-145 2014.

Łukasz Szczepkowicz, B. P.: Quality Evaluation Of Selected Tools To Automate Application Testing, Journal Computer Science Institute, Vol. 36b, Pp. 20-618 2018.

Colorado Rivera, L. P.: Automatización De Pruebas Funcionales, Un Complemento Para La Calidad Del Software, Especialización En Gerencia Integral De Proyectos Universidad Militar Nueva Granada Facultad De Ingeniería, 2020.

Wynne, M. H., A. The Cucumber Book. Behaviour-Driven Development For Testers And Developers. Dallas, Texas: Pragmatic Programmers, 2012. 978-1934356807.

Fontela, M. C.: Estado Del Arte Y Tendencias En Test-Driven Development, Facultad De Informática Universidad Nacional De La Plata 2011.

De Florinier, D. A., G. The Secret Ninja Cucumber Scrolls: Strictly Confidential. Londres, Reino Unido, 2011.

Delgado, M. M., A, Larrosa, D, Verona, S. Fernández, P.: Model For Automatic Generation Of Search- Based Early Test. Computación Y Sistemas, 2017, Vol. Vol 21, No. No 3. Issn: 2007-9737

Polo, M. U. "Towards Green Software Testing". En: Green In Software Engineering. C. Calero, M. P. Switzerland: Springer International Publishing, 2015.

Tzancoff B. Diaz, Claudia -Rodríguez,Anahí -Soria, V.: Usando Jmeter Para Pruebas De Rendimiento, Linti. Fac. De Informática, Universidad Nacional De La Plata.

Ranadhan, R.: Pembuatan Functional & Technical Requirements Aplikasi Kreasi Bri Dan Testing Aplikasi Ceria Bri Menggunakan Katalon Studio, Teknik Informatika Fteic-Its Institut Teknologi Sepuluh Nopember 2020.

Holmes And Kellog, M.: Automating Functional Tests Using Selenium, 2006.

Kaur And Gupta, D. G.: Comparative Study Of Automated Testing Tools: Selenium, Quick, Test Professional And Testcomplete, Journal Of Engineering Research And Applications, Vol. 3, Pp. 1739-1743, 2013.

Chopra, N. U.-V.: Design And Implementation In Selenium Ide With Web Driver, International Journal Of Computer Applications Vol. 46, 2012.

Conflicto de interés

El autor autoriza la distribución y uso de su artículo.

Contribuciones de los autores

1. Conceptualización: Nombre y Apellidos del autor.
2. Curación de datos: Nombre y Apellidos del autor
3. Análisis formal: Nombre y Apellidos del autor
4. Adquisición de fondos: Nombre y Apellidos del autor

5. Investigación: Nombre y Apellidos del autor
6. Metodología: Nombre y Apellidos del autor
7. Administración del proyecto: Nombre y Apellidos del autor
8. Recursos: Nombre y Apellidos del autor
9. Software: Nombre y Apellidos del autor
10. Supervisión: Nombre y Apellidos del autor
11. Validación: Nombre y Apellidos del autor
12. Visualización: Nombre y Apellidos del autor
13. Redacción – borrador original: Nombre y Apellidos del autor
14. Redacción – revisión y edición: Nombre y Apellidos del autor

Financiación

Universidad Tecnológica de La Habana “José Antonio Echeverría”, CUJAE