A Comparative Study of GUI Automated Tools for Software Testing

Peter Sabev
Department of Informatics and Information Technologies
"Angel Kanchev" University of Ruse
Ruse, Bulgaria
e-mail: psabev@uni-ruse.bg

Prof. Katalina Grigorova
Department of Informatics and Information Technologies
"Angel Kanchev" University of Ruse
Ruse, Bulgaria
e-mail: kgrigorova@uni-ruse.bg

***Abstract*—Nowadays, a main resort for delivering software with good enough quality is to design, create, implement and maintain test cases that are executed automatically. This could be done on many different levels, however graphical user interface (GUI) testing is the closest one to the way the real user interacts with the software under test (SUT). The aim of this paper is to determine the most popular GUI automated tools for software testing among a list of 52 candidates and compare them according to their features, functional and non-functional characteristics.**

Keywords-GUI; software; quality assurance; QA; automated testing; test automation; testing tools; UI; GUI; tests; Selenium; RFT; UFT; TestComplete; Ranorex; OATS.

I. INTRODUCTION

Testing is an essential activity to ensure quality of software systems. Automating the execution of test cases against given software or system can greatly improve test productivity, and save time and costs.

However, many organizations refuse to use test automation or have failed on implementing it because they do not know how to deal with the implementation of a test automation strategy.

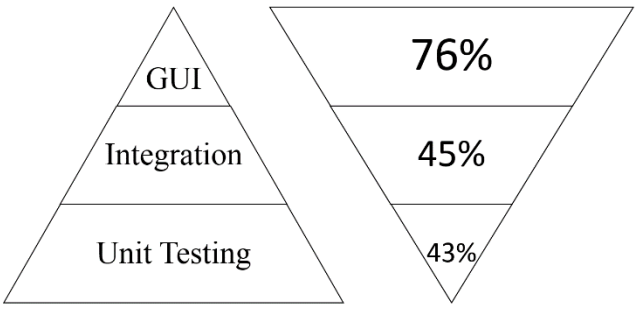

Figure 1. The ideal test automation pyramid on the left and the reversed reality on the right.

In 2009, Mike Cohn proposed the test automation pyramid (Fig. 1) that has become a best practice in the software industry. According to the pyramid, unit testing should be the majority of tests, creating foundation of the testing strategy, later expanded by service-level integration tests and finished by GUI automated tests [1]. GUI tests are time-consuming, harder to maintain, thus they are placed on the top of the pyramid, aiming to do as little user interface tests as possible. However, the reality shows a totally reversed situation. In many companies, because of the isolation of the role of QA engineers and tasking them to write GUI tests only, the ration of GUI tests to unit tests is inverse. Although it is not possible to collect ratio for test distribution in each project, reports from 2016 show that unit testing is done in only 43% of the software companies, while 45% of them do integration testing, and GUI test automation is done in 76% of the companies. 60% of quality assurance engineers claim to design, implement and maintain scripted testing, and 39% claim to do user simulations. The report also shows that 94% of the software engineers consider functional automation and scripting important, and 67% find automation tools challenging or extremely challenging [2].

All the above indicates that choosing a satisfactory GUI automated testing tool or framework is very important task, and a challenging problem to solve at the same time. The incorrect choice of proper GUI testing tool may result significant loss of time and money, and may even lead one to be unable to automate their GUI testing entirely. This paper conducts a comparative analysis of 52 state-of-the-art tools and provides comparison tables that could direct towards the most suitable tool according to their requirements.

In the next section, a methodology for creating a comprehensive list of tools is described. In Section III, the list of GUI tools is filtered by popularity and maturity. In Section IV, a final assessment is made for the top candidates, giving them score in eight different categories. The outcome of that assessment is shown in Section V, and as the scores are quite close, details for each of the finalist are given in Section VI. A conclusion based on this paper is made in Section VII.

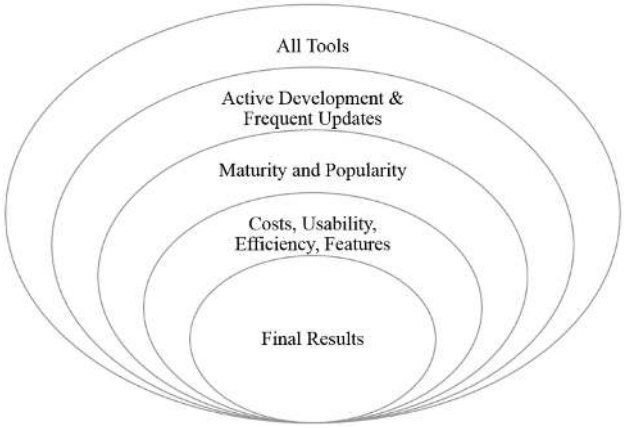

Figure 2. Automation Tools Selection Criteria

II. METHODOLOGY

A comprehensive list of 52 automated testing tools is created (Table I), based on [3]-[5] and the information available in the websites listed in the table itself.

The list consists of both free and proprietary tools that are web-based or work at least on one of the following operating systems: Windows, Linux or MacOS. Only proprietary tools with available demo versions are considered. Then some tools are discarded from the list, according to the criteria shown on Fig. 2.

The active development of a testing tool is very important, so tools with no active development after 2015, as well as all deprecated tools are later discarded from the list.

As Table I shows, only 30 tools have active development after 2015, and those tools were listed in Table II. The percentage of active, inactive and discontinued development is shown on Fig. 3.

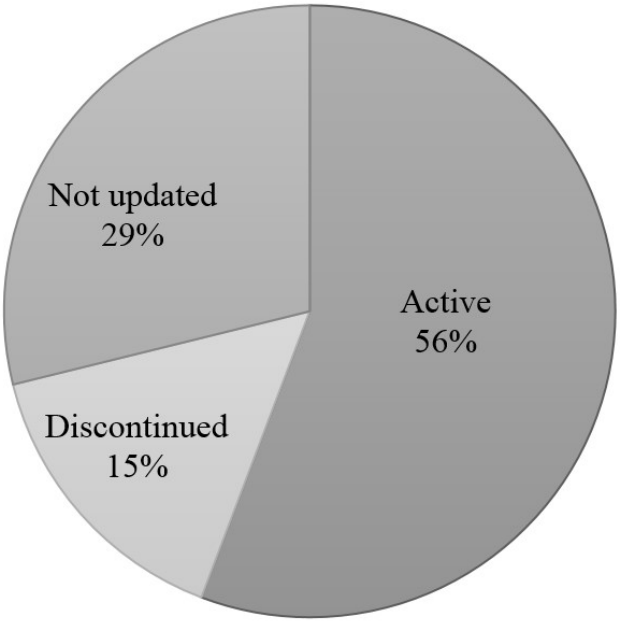

Figure 3. Distribution of active, discontinued and inactive development of automated GUI testing tools.

Only the 10 most popular tools that are mature enough made it to the final stage where a comparison against cost effectiveness, functional and non-functional characteristics is made. This is further explained later in this paper.

III. MATURITY AND POPULARITY

It is very important that automated testing tools are mature enough, being on the market for at least 3 years, as it takes time until the tools are polished according to the needs of their users. All of the tools remaining are on the market since 2014 or earlier, with SilkTest being the oldest tool with active development in this study (since 1999). It is also very important for a testing tool to be popular in the software engineering industry (so as many professionals as possible know about the product, its features and how to use it). It should be popular also among the scientific researchers (so innovations are presented continuously) and the QA community (as people need to help each other, contribute and give suggestions for improvements). That is why the following criteria are chosen to determine tools popularity:

- Google Results (GR Rank) – Google Search [6] is conducted with the name of the tool and the vendor together. When the product is community-driven or there is ambiguous tool name (e.g., Selenium), the phrase "testing tool" is added to the search. All results are recorded, the list is then sorted by the number of the search results, and finally ranking is assigned and recorded in the GR Rank column. The search results were returned using Google Chrome in incognito mode with Bulgarian IP address. Last, but not least, it is hard, if not impossible task to distinguish positive and negative mentions in the search results. Popularity, however, consists of both, i.e., if one knows about a given tool but they do not like it, the tool is still popular;
- To assess the popularity in the scientific community, a search similar as above is performed in Google Scholar (GS Rank) [7] and ResearchGate (RG Rank) respectively [8];
- Website popularity is assessed according to Alexa website rank [9]. Although this rank is focused towards the website and respectively the software vendor, popular vendors are expected to be more reliable and software to have longer support lifecycle. The rankings are recorded under A Rank column;
- Wikipedia page views are measured using a web statistics tool [10] (0 is written for the tools with no dedicated Wikipedia page), and ranking is recorded under W Rank column.

The different criteria may have different importance for the different researches, so the popularity can be calculated in many ways. For the general case of this study, an average of the GR, GS, RG, A and W columns is calculated and recorded under the Popularity Index column and Table II is then sorted according to that criteria. The top 10 tools based on their popularity index are considered for the next stage.

IV. FINAL ASSESSMENT

In the final stage, each of the top 10 candidates is assessed in eight different categories. In each category, maximum 5 points are given, forming a maximum of 40 points per tool. The scores given to some of the categories are not normalized intentionally, to allow adding future tools without changing the scoring system.

A. Popularity (P)

Popularity assessment is described above already. 5 points are given to tools with popularity index below 3.0; 4 points when the index is from 3.1 to 6.0; 3 points - 6.1 to 7.5; 2 points - 7.6 to 9.0; 1 point - 9.0 to 15 and no points are given for popularity index that is more than 15. As already mentioned above, it is a challenging task to determine popularity objectively and with good precision.

The main idea is to give similar points according to tools popularity index. That is why the border values are chosen in a way to provide equal distribution of points for relatively equal segments of tools with similar popularity indexes.

B. Licensing Costs (LC)

Licensing costs are very important factor in many software companies. Thus, the maximum of 5 points is given to the free tools, 4 points are given to tools that cost under $1000 per single license, 3 points - for tools with single license between $1000 and $2000; 2 points - from $2001 to $5000; 1 point - from $5000 to $10000, and no points are given for license above $10000. For period-based licenses, the period considered is 3 years. Again, border values are chosen with equal distribution of points in mind.

C. Installation, Configuration and Online Documentation Availability (IC)

First experience that a given user has with an automation testing tool is its installation and configuration. If the tool can be installed, configured and a simple application can be run within 60 minutes, the tool is considered easy to install and configure, so it receives 2 points. Online documentation availability is also considered during that process and additional point is given for that. The last 2 points are given if the tool supports at least one (1 point for one, 2 points for more than one) system for continuous integration and continuous delivery (CI/CD). The list of CI/CD systems was taken from Wikipedia [11].

D. Usability (U)

According to ISO 9241-11, usability is the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use [12]. System Usability Scale (SUS) questionnaire [13] was filled by 10 different software QA engineers at different level of expertise. Points are given as follows: 5 points are given for SUS score between 85 and 100; 4 points – for SUS score from 73 to 84; 3 points – from 52 to 72; 2 points – from 39 to 51; 1 point – from 26 to 38. No points are given for SUS score of 25 or less.

E. Programming Skills Needed (PS)

Programming skills and the number of programming languages available are also very important factor when assessing GUI automation test tool. If the tests can be created and executed without (or minimum) programming skills, this means that much more people would be able to use the tool and the final costs is expected to be less. 5 points were given to tools which require no programming skills; 4 points when simple programming scripts are needed, and the tool gives a choice of programming language; 3 points are given when no choice of programming language is available. For complex scripts, 2 points are given where choice of programming language is available and 1 point if there is no such choice. No points are given where scripts are too complex and even professional programmer is not able to automate the test cases needed. Of course, it is not necessarily true that the maintenance of tools requiring programming is more expensive compared to record-and-play ones, and this assessment is handled by the next metric.

F. Recording and Playback of Test Scripts (RP)

Once installed, recording and replay of test script becomes a major part of tool usage. Inaccurate recording and replay usually causes more maintenance effort. Data-driven approach separates the logic from the test data, and this makes the maintenance easier. Thus, one point is given to all tools that support data-driven testing. When recording is possible via both scripts and UI, the playback is easy and no problems are found, the maximum of 4 additional points is given. One point less is given when there is only one option for recording and replay has no problems. Same is done when both options for recording are available and minor problems are found while replaying. When there is only one option for recording and minor problems are found, 2 points are given. If there are major problems, 1 point is given if there is a workaround, and 0 points – when there is no workaround.

G. Efficiency (E)

Quick test execution is also very important for a test tool, especially when there are many test cases to be automatically executed. A simple set of 4 test cases is recorded on the different tools, testing Windows Calculator and Google Calculator (network delay times are removed), exercising addition, subtraction, multiplication and division. 5 points are given when the whole execution takes less than 5 seconds. From there on, 1 point is subtracted for doubling the execution period, i.e. 4 points are given for execution times from 5 to 10 secs; 3 points – from 11 to 20 secs; 2 points – from 21 to 40 secs; 1 point – 41 to 90 secs. No points are given for test execution that take more than 90 secs.

H. Quality of Reports (QR)

Last but not least, test reporting provides important information on how the test execution went. 5 points are awarded for automatically generated and highly configurable test reports, 4 points – for reports that are automatically generated but not configurable; 3 points – for reports that can be manually created or easily integrated; 2 points – when there is at least possibility to integrate the tool with other reporting tools or systems; 1 point – if such integration is not supported but possible with workarounds. No points are given if such integration is not possible.

TABLE I. FULL LIST OF GUI AUTOMATED TOOLS FOR SOFTWARE TESTING CONSIDERED

Name	Developer or Vendor	Website (URL)	Latest version	OS[1]	Supported Languages	License	Demo	St[2]	Last Update
Abbot Java GUI Test Framework	Timothy Wall	http://abbot.sourceforge.net/	1.3.0	WLM	Java	EPL	N/A	NU	2015
App Test	AppPerfect	http://www.appperfect.com/products/app-test.php	14.5	WLM	Java	$299-$399 per user	15 days	Act	2016
Ascential test	Zeenyx Software, Inc.	http://www.zeenyx.com/AscentialTest.html	6	Web	Java, .NET	Proprietary	On request	Act	2016
AutoIt	AutoIt	https://www.autoitscript.com/site/autoit/	3.3.14.2	W	Own BASIC-like language	Freeware (closed source)	N/A	NU	2015
Coded UI Test	Microsoft	https://docs.microsoft.com/en-us/visualstudio/test/use-ui-automation-to-test-your-code	(part of Visual Studio)	W	.NET	$1200 per user	No	Act	2016
CubicTest	CubicTest	https://github.com/cubictest/	2.21.0	WLM	Java	EPL	N/A	Dis	2012
Dojo Objective Harness	Dojo Foundation	https://dojotoolkit.org/	1.11.3	Web	JavaScript	AFL	N/A	Act	2017
eggPlant Functional	Test Plant Ltd	http://www.testplant.com/	17.0.2	WLM	Java, .NET, C#, Ruby, C++, Python	Proprietary	5 days	Act	2017
eZscript	Universal Test Solutions	http://www.uts-global.com/eZscript.html	0.375	W	XML, keyword driven	Proprietary	On request	NU	2010
Fake	Celestial Teapot	http://fakeapp.com/	1.9.1	M	AppleScript, JavaScript	$30	Freemium	Act	2016
FEST	Google Code / Atlassian	https://code.google.com/archive/p/fest/wikis/Github.wiki	0.30	WLM	Java	Freeware (open source)	N/A	Dis	2013
FitNesse	Community-driven	http://fitnesse.org/	20160618	WLM	Java, Python, C#	Freeware (open source)	N/A	Act	2016
Gauge	Thought Works, Inc.	http://getgauge.io/	0.7.0	WLM	.NET, Java, Ruby	GPLv3	N/A	Act	2017
Google Test	Google Inc.	https://github.com/google/googletest	1.8.0	WLM	C++	Freeware (open source)	N/A	Act	2016
GTT (GUI Test Tool)	Prof. Woei-kae Chen	http://gtt.sourceforge.net/	3.0	WLM	Java	Freeware (open source)	N/A	Dis	2009
IcuTest	NXS-7 Software Inc	http://www.nxs-7.com/icu/	1.0.7	W	.NET	Proprietary	N/A	Dis	2010
iMacros	Ipswitch, Inc.	http://imacros.net/	11.2	WWeb	JavaScript	Proprietary	30 days	Act	2016
IronAHK	Community-driven	https://github.com/polyethene/IronAHK	0.7	W	.NET	Freeware (open source)	N/A	NU	2010
Jameleon	Community-driven	http://jameleon.sourceforge.net/	3.3	WLM	Java, XML	Freeware (open source)	N/A	NU	2013
Jubula	Eclipse & BREDEX GmbH	http://www.eclipse.org/jubula/	8.4.0	WL	Java, Swing, HTML	Freeware (open source)	N/A	Act	2017
Linux Desktop Testing Project	Community-driven	https://ldtp.freedesktop.org/	3.5.0	WLM	Java, .NET, Python, Ruby, Perl, Clojure	GNU LGPL	N/A	NU	2013
Marathon	Jalian Systems	https://marathontesting.com/	5.0.0.0	WLM	Java, Swing, Ruby	$1480 per user	30 days	Act	2016
Maveryx	Maveryx srl	http://www.maveryx.com/	1.5	WLM	Java	2000 EUR per year	Freemium	Act	2016
Oracle Application Testing Suite	Oracle	http://www.oracle.com/technetwork/oem/app-test/index.html	12.5.0.3.0	Web	Own, OpenScript (Java)	Proprietary	Freemium	Act	2016

[1] Supported Operating System (OS): W – Windows, L – Linux, M – MacOS, Web – Web-Based Applications
[2] Update Status: NU – Not updated since 2015, Dis – Officially discontinued, Act - Active

Pounder	Community-driven	http://pounder.sourceforge.net/	0.95	WLM	Java	GNU LGPL	N/A	NU	2002
QA Liber	Community-driven	http://qaliber.org/	1.0	W	.NET	GPLv2	N/A	NU	2011
QF-Test	Quality First Software GmbH	https://www.qfs.de/en.html	4.1.1	WLMWeb	Java, Swing	2000 EUR per user	30 days	Act	2016
Ranorex	Ranorex GmbH	http://www.ranorex.com/	6.2.0	WWeb	.NET	$2590/user	30 days	Act	2016
Rational Functional Tester	IBM	http://www-03.ibm.com/software/products/en/functional	8.6.0.7	WL	Java, VBScript	$6820/user	30 days	Act	2016
RCP Testing Tool	Eclipse	https://eclipse.org/rcptt/	2.2.0	WLM	Eclipse Common Language	Freeware (open source)	N/A	Act	2017
RIATest	Cogitek Inc.	http://www.cogitek.com/riatest.html	6.2.6	WM	Own, RIAScript	Proprietary	30 days	NU	2015
Robot Framework	Community-driven	http://robotframework.org/	3.0	WLM	Java	Apache	N/A	NU	2015
Sahi	Tyto Software	http://sahipro.com/	6.3.2	Web	Java	$695 per user/year	30 days	Act	2016
Selenium	Community-driven	http://www.seleniumhq.org/	3.0.1	Web	Java, .NET, JavaScript, Python, Ruby, PHP, Perl, R, Objective C, Haskell	Apache	N/A	Act	2016
SikuliX	MIT	http://sikulix.com/	2.0.0	WLMWeb	Ruby, Python, Java, Jython	MIT	N/A	NU	2015
SilkTest	Micro Focus Int.	https://www.microfocus.com/products/silk-portfolio/silk-test/	17.5	WL	Java, .NET, own (C++ like)	Individual offer ($5K-9K)	45 days	Act	2016
Squish GUI Tester	froglogic GmbH	https://www.froglogic.com/squish/	6.2	WLM	Keyword-driven	4000 EUR per user	60 days	Act	2016
SWAT	Community-driven	https://sourceforge.net/projects/ulti-swat/	4.1	WWeb	.NET	GPLv2	N/A	Dis	2012
SWTBot	Eclipse	http://www.eclipse.org/swtbot/	2.5.0	WL	Java	Freeware (open source)	N/A	Act	2016
Telerik Test Studio	Progress	http://www.telerik.com/teststudio	2016.4.1208.2	WWeb	HTML, .NET, JavaScript, Ruby, PHP, own (NativeScript)	$2499	30 days	Act	2016
Tellurium	Grant Street Group	http://www.te52.com/	N/A	Web	Java, Perl Python, Ruby	Freeware (closed source)	N/A	Act	2016
Test Complete	SmartBear Software	https://smartbear.com/product/testcomplete/	42804	WWeb	JavaScript, Python, VBScript, JScript, Delphi, C++, C#	3730 EUR	30 days	Act	2016
Testing Anywhere	Automation Anywhere, Inc.	https://www.automationanywhere.com/testing	9.3	W	Keyword-driven	Proprietary	On request	NU	2015
TestPartner	Micro Focus Int.	https://www.microfocus.com/products/silk-portfolio/silk-testpartner/	6.3.2	W	.NET	Proprietary	45 days	Dis	2014
TestStack.White	Community-driven	https://github.com/TestStack/White	0.13	W	.NET	Freeware (open source)	N/A	NU	2014
Tosca Automate UI	Tricentis GmbH	https://www.tricentis.com/resource-assets/tosca-automate-ui/	10.1	W	VBScript	Proprietary	14 days	Act	2017
Twist	Thought Works, Inc.	https://www.thoughtworks.com/products/twist-agile-testing	Unknown	WLM	Java	Proprietary	N/A	Dis	2014
UI Automation Powershell Extensions	Community-driven	https://uiautomation.codeplex.com/	0.8.7	W	PowerShell	Freeware (open source)	N/A	NU	2014

Unified Functional Testing (UFT)	HP Enterprise	https://saas.hpe.com/en-us/resources/uft	12.54	W	Own, keyword driven, VBScript	$3200 per user/year	30 days	Act	2016
VisualCron	NetCart	http://www.visualcron.com/	8.2.3	W	PowerShell, SQL, Batch	$149 per server/year	45 days	Act	2016
Watir	Community-driven	https://watir.com/	6.1	Web	Ruby	MIT	N/A	Act	2017
WinRunner	HP Enterprise	https://softwaresupport.hpe.com/document/-/facetsearch/document/KM01033448	9.2	W	Own, scripting	Proprietary	N/A	Dis	2008
Xnee	Community-driven	https://www.gnu.org/software/xnee/	3.19	L	X11 protocol used	GPLv3	N/A	NU	2014

TABLE II. FILTERED TOOLS SORTED BY POPULARITY INDEX

Tool Name	Since Year	Google Results	GR Rank	Google Scholar Results	GS Rank	Research Gate Results	RG Rank	Alexa Site Rank	A Rank	Wikipedia Views	W Rank	Popularity Index
Selenium	2006	1000000	1	1490	1	580	1	24147	10	21637	1	**2.8**
Rational Functional Tester	2007	425000	3	518	4	8	3	614	4	767	12	**5.2**
Google Test	2008	114000	12	437	5	5	6	61	2	2431	4	**5.8**
Unified Functional Testing	2000	418000	4	100	10	3	8	3629	8	4416	2	**6.4**
TestComplete	1999	553000	2	112	9	3	8	32279	11	1326	5	**7**
FitNesse	2009	328000	5	690	2	17	2	586115	21	1162	6	**7.2**
Coded UI Test	2010	73700	13	50	16	6	4	40	1	1036	7	**8.2**
SilkTest	1994	191000	7	615	3	3	8	61962	13	792	10	**8.2**
Ranorex	2007	125000	10	169	7	2	12	161122	15	901	9	**10.6**
Oracle Application Testing Suite	2008	190000	8	41	18	2	12	348	3	604	13	**10.8**
iMacros	2001	3950	23	201	6	4	7	35068	12	942	8	**11.2**
Watir	2011	125000	11	138	8	6	4	881518	24	789	11	**11.6**
Jubula	2012	4870	22	91	11	3	8	2956	7	23	22	**14**
RCP Testing Tool	2014	143000	9	88	12	0	17	2956	5	0	30	**14.6**
SWTBot	2001	51200	15	70	13	2	12	2956	6	0	29	**15**
Telerik Test Studio	2002	29100	16	22	21	0	17	4655	9	544	15	**15.6**
Dojo Objective Harness	2011	747	29	10	24	0	17	86069	14	3277	3	**17.4**
eggPlant Functional	2013	24200	17	23	20	0	17	522174	18	489	16	**17.6**
QF-Test	2001	8540	20	68	14	2	12	2001348	27	241	18	**18.2**
Gauge	2014	12900	19	65	15	0	17	526578	19	112	21	**18.2**
Maveryx	2010	256000	6	16	23	0	17	7029069	30	258	17	**18.6**
Tricentis Tosca	2011	13000	18	24	19	2	12	281142	17	0	27	**18.6**
Sahi	2010	3210	25	22	22	0	17	534433	20	555	14	**19.6**
VisualCron	2005	69300	14	5	27	0	17	910942	25	175	20	**20.6**
App Test	2003	2050	28	42	17	0	17	877243	23	0	25	**22**
Marathon	2005	5780	21	7	25	0	17	3542542	28	0	23	**22.8**
Tellurium	2010	3320	24	7	26	0	17	630725	22	0	26	**23**
Squish GUI Tester	2003	2820	26	5	28	0	17	202006	16	0	28	**23**
Fake	2010	2440	27	5	29	0	17	1816998	26	0	24	**24.6**
Ascentialtest	2006	524	30	1	30	0	17	4692844	29	197	19	**25**

V. FINAL RESULTS

After the final assessment in the eight different categories above, Table III containing the final rankings is produced:

TABLE III. FINAL CANDIDATES ASSESSMENT RESULTS

№	Tool Name	P	LC	IC	U	PS	RP	E	QR	TOTAL
1	UFT	5	5	4	3	4	2	3	2	29
2	Selenium	4	1	3	4	4	4	4	5	28
3	Ranorex	2	3	4	3	3	4	4	2	26
4	CUITs	1	2	3	4	4	5	3	4	25
5	Google Test	4	5	3	1	1	2	5	1	22
5	TestComplete	3	5	3	2	2	2	3	1	22
7	FitNesse	3	2	3	3	3	3	2	3	21
8	SilkTest	2	1	3	3	4	2	3	2	20
9	OATS	4	1	2	3	3	1	3	2	20
10	RFT	1	1	3	3	3	3	2	4	19

VI. FINAL CANDIDATES

The top 10 GUI automation tools that are taken into the final comparison are quite different in many aspects – environment, installation, usage, test script creation and maintenance, etc. That is why this section is dedicated on providing more detailed description of each finalist, so one could pick up the best candidate according to their specific needs:

A. Selenium

Selenium is nowadays the most popular software testing framework for web applications. Selenium is portable and provides a record/playback tool for authoring tests without learning a test scripting language (Selenium IDE). It also provides a test domain-specific language (Selenese) to write tests in a number of popular programming languages, including C#, Groovy, Java, Perl, PHP, Python, Ruby and Scala. Selenium WebDriver accepts commands (sent in Selenese, or via a Client API) and sends them to a browser. It does not need a special server to execute tests. Instead, the WebDriver directly starts a browser instance and controls it. However, Selenium Grid can be used with WebDriver to execute tests on remote systems. Selenium allows parallel executions, has multi-platform and multi-browser support (although there are some issues with Safari and Internet Explorer). Selenium supports a variety of CI/CD tools, however some programming may be needed for full setup. A big amount of programming and setup is needed to integrate with report generation tools or database (for data-driven testing). From recording and playback perspective, there is no option to run the test from a point or state of application, so tests need to be started from the very beginning each time. Test execution speed highly depends on the locator used, e.g., Xpath selectors are much slower compared to getting elements by their ID.

B. Rational Functional Tester

IBM Rational Functional Tester (RFT) provides automated testing capabilities for functional, regression, GUI, and data-driven testing. Installation is straight forward. The RFT can generate VBScript and Java statements, requiring some programming experience. Test execution is generally stable, but occasionally it has memory issues, which can be solved easily. During playback, Rational Functional Tester uses the Object Map to find and act against the application interface. However, during development it is often the case that objects change between the time the script was recorded and when a script was executed. For example, testing with multiple values selected using the Shift key pressed does not work. RFT supports data driven commands to generate different test cases, however the expected outputs need to be manually entered. RFT allows one script to call another script, so redundant activities are not repeated. However, scripts quickly become too long and hard to maintain. From reporting point of view, RFT supports results logs containing a lot of information, making hard to find the data really needed. It also supports customized reports but integration takes a lot of time. CI/CD integration is supported only for IBM products.

C. Google Test (with Google Mock)

Google Test (also known as Google C++ Testing Framework) is a unit testing library for the C++ programming language, based on the xUnit architecture. Google Test cannot be used for GUI automation tests as standalone tool. In this study, it is used together with Google Mock, so one can create mock classes trivially using simple macros, supporting a rich set of matchers and actions, and handling unordered, partially ordered, or completely ordered expectations. The framework uses an intuitive syntax for controlling the behavior of a mock, however it has been intended to support unit tests rather than GUI, so most testers find its installation, configuration and coding too complex. Google Test is good to consider for a team of highly skilled developers in test.

D. Unified Functional Testing (UFT)

HPE Unified Functional Testing (UFT) software, formerly known as HP QuickTest Professional (QTP), provides functional and regression test automation for software applications and environments. UFT is targeted at enterprise QA, supporting keyword and scripting interfaces and features a graphical user interface. The keyword view allows a novice tester to easily work with the tool. However, UFT often has problems recognizing customized user interface objects and other complex objects which need to be defined as virtual objects, requiring technical expertise. UFT can be extended with separate add-ins, e.g., support for Web, .NET, Java, and Delphi. UFT runs primarily on Windows, relying on obsolete ActiveX and VBScript which is not an object-oriented language. CI/CD integration is limited, as Test Execution engine is combined with the GUI Test Code development IDE. It is not possible to run the tests independent of HPE Unified Functional Testing. High licensing costs often mean that the tool is not widely used in

an organization, but instead is limited to a smaller testing team, encouraging testing to be performed as a separate phase rather than a collaborative approach (as advocated by agile development processes). Test execution is quick, although it causes high hardware load.

E. TestComplete

TestComplete is a functional automated framework for Microsoft Windows, Web, and smartphone applications, providing separate modules for each platform. Tests can be recorded, scripted or manually created with keyword-driven operations and used for automated playback and error logging. The tool records only the key actions necessary to replay the test and discards all unneeded actions, supporting data-driven testing. Biggest product drawbacks are crashes, hangs and long waiting times (especially for DOM objects), as well as problems with reading XPath values for some browsers. Regular expressions and descriptive programming are not supported. TestComplete has good integration with CI/CD and reporting tools, although it might require technical expertise.

F. FitNesse

FitNesse is an integrated framework consisting of web server, a wiki and an automated testing tool for software, focused on GUI acceptance tests. FitNesse was originally designed as a highly usable interface around the Fit framework. As such, its intention is to support an agile style of black-box testing acceptance and regression testing. Installation is simple. Tests are described in wiki as decision tables, with coupled inputs and outputs. The link between those tables and the system under test is made by a piece of Java code called a fixture. FitNesse comes with its own version control but also can be integrated with external one. People with no programming skills are unable to use FitNesse, except adding or maintaining test cases in the wiki. The major drawbacks are that those tests are often limited. Also, recent FitNesse releases have issues with backward compatibility and lack of error messages or feedback on what during test execution went wrong.

G. Microsoft Coded UI Tests (CUITs)

Automated tests in Microsoft application that go through its user interface are known as coded UI tests (CUITs). These tests include functional testing of the UI controls. CUITs are available as separate project in Microsoft Visual Studio (VS). Recording and playback actions can be easily done with Microsoft Test Manager or VS, and scripts can be maintained on the fly. Data-driven testing is supported for any data source supported by .NET framework. CUITs seamlessly integrates with Team Foundation Server (TFS), supports Application Lifecycle Management (ALM) and can even execute web-based test on Internet Explorer browser only. CUITs are very easy to use by people who are familiar with VS development but not an option for those who are not.

H. SilkTest

SilkTest is focused to automated function and regression testing of enterprise applications and consists of two parts: host that contains all the source script files and agent that translates the script commands into GUI commands. Separation of test design and test implementation, together with keyword-driven framework, object-oriented approach and both ability to capture objects from UI or use descriptive programming, makes the tool a great candidate to consider. SilkTest was originally developed by Segue Software, acquired by Borland in 2006, which was also acquired by Micro Focus International in 2009. Those acquisitions, and the fact this is the oldest tool among the finalists (more than 23 years), logically brings some issues, e.g., GUI interface is not modern and looks too complex to non-developers. While the installation is smooth, recording mode generates code that is hard to read, and sometimes there is no other way to interact with specific objects other than coordinate-based. Also, online documentation needs improvement.

I. Ranorex

Ranorex is a GUI test automation framework for desktop, web-based and mobile applications that also supports unit tests. Installation is straightforward, and there is plenty of online documentation and video tutorials. Element recognition is very reliable (XPath and image-based) and both record/replay tool and descriptive programming in C# and VB.NET are supported. Test suites generate executable files that can be easily run by launching the .EXE file where needed. Tests can be recorded by people with no programming skills, and logs are easy to navigate through. However, test execution is unstable at times, especially on remote or virtual machines. Also, the logs are not in common format, so they need to be additionally parsed to include in most CI/CD systems. Another drawback is that no additional plugins are supported.

J. Oracle Application Testing Suite (OATS)

OATS is a web-based comprehensive, integrated testing solution. It has excellent co-relation with all Oracle applications and uses a functional testing module called OpenScript. Same script can be run on different instances. The report that is generated is quite detailed and useful but becomes too big. Performance is slower compared to the most tools, browser often runs out of memory, and significant programming knowledge is needed. Data-driven testing is supported using a feature called Data bank.

VII. CONCLUSION

There is no perfect GUI test automation tool. The fact that even the top scoring tool achieved only 29 out of 40 points shows that each of these tools has its drawbacks and room for improvement. Also, the difference between the first and the last of the finalist tools is just 10 points, which suggests that one should take all factors under consideration when choosing GUI automation tools.

This paper can be extended in future by adding more automation tools, adding more assessment factors and modifying the methodology for the existing ones, but the key

points in creating a successful automated tool for software testing on GUI level are clear: such a tool needs to support both engineers with no programming skills (via Record/Replay features, understandable GUI, image-based recognition) and engineers with good programming skills (with Java and .NET being the most popular programming platforms). The tool should support additional plugins, CI/CD integration, reporting tools and high customization on different levels. The proper replaying of recorded scripts is a must, and the maintainability of test scripts is crucial. Last, but not least, tool maturity and popularity, good support, online documentation and big community are important additions for a complete product.

The findings of this paper may be valuable for the scientific community and the industry as a reference list for educational purposes or as baseline for picking the right testing tool. The initial list could produce completely different results if comparison is made against different criteria, according to specific needs of individual or business, project, environment and budget.

ACKNOWLEDGMENT

This work is supported by the National Scientific Research Fund under the contract ДФНИ-И02/13.

REFERENCES

[1] M. Cohn, Succeeding with Agile: Software Development Using Scrum, Upper Saddle River, NJ: Addison Wesley, 2009, pp. 312-314.

[2] PractiTest, Tea Time with Testers, "State of Testing Report," PractiTest, Rehovot, Israel, 2016.

[3] Wikimedia Foundation, "Comparison of GUI testing tools," [Online]. Available: https://en.wikipedia.org/wiki/Comparison_of_GUI_testing_tools. [Accessed 14 01 2017].

[4] Y. Ben-Hur, "QA Testing Tools: All About Software Testing Tools," [Online]. Available: http://qatestingtools.com/compare-gui-testing-tools. [Accessed 14 01 2017].

[5] A. B. C. Brahim, "Evaluation of Tools of automated testing for Java/Swing GUI," Paris, 2014.

[6] Google Inc., "Google Search," Google Inc., [Online]. Available: https://www.google.com. [Accessed 14 01 2017].

[7] Google Inc., "Google Scholar," Google Inc., [Online]. Available: https://scholar.google.com/. [Accessed 14 01 2017].

[8] ResearchGate, "ResearchGate," researchgate.net, [Online]. Available: https://www.researchgate.net/home. [Accessed 14 01 2017].

[9] Alexa Internet, Inc. , "Find Website Traffic, Statistics, and Analytics," Alexa Internet, Inc. , [Online]. Available: http://www.alexa.com/siteinfo. [Accessed 14 01 2017].

[10] Wikimedia Foundation, "Wikipedia:Web statistics tool," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Wikipedia:Web_statistics_tool. [Accessed 14 01 2017].

[11] Wikimedia Foundation, "Comparison of continuous integration software," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Comparison_of_continuous_integration_software. [Accessed 15 01 2017].

[12] International Organization for Standardization, ISO/DIS 9241-11.2: Ergonomics of human-system interaction - Part 11: Usability: Definitions and concepts, Geneva: ISO, 2016.

[13] J. Brooke, "SUS-A quick and dirty usability scale," Usability evaluation in industry, vol. 189, no. 194, pp. 4-7, 1996.