A Survey on Testing Distributed and Heterogeneous Systems: The State of the Practice

Bruno Lima[1,2](✉) and João Pascoal Faria[1,2]

[1] INESC TEC, FEUP Campus, Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal
{bruno.lima,jpf}@fe.up.pt

[2] Faculty of Engineering, University of Porto, Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal

Abstract. Distributed and heterogeneous systems (DHS), running over interconnected mobile and cloud-based platforms, are used in a growing number of domains for provisioning end-to-end services to users. Testing DHS is particularly important and challenging, with little support being provided by current tools. In order to assess the current state of the practice regarding the testing of DHS and identify opportunities and priorities for research and innovation initiatives, we conducted an exploratory survey that was responded by 147 software testing professionals that attended industry-oriented software testing conferences. The survey allowed us to assess the relevance of DHS in software testing practice, the most important features to be tested in DHS, the current status of test automation and tool sourcing for testing DHS, and the most desired features in test automation solutions for DHS. Some follow up interviews allowed us to further investigate drivers and barriers for DHS test automation. We expect that the results presented in the paper are of interest to researchers, tool vendors and service providers in this field.

Keywords: Software testing · Distributed systems · Heterogeneous systems · Systems of systems · State of the practice

1 Introduction

Due to the increasing ubiquity, complexity, criticality and need for assurance of software-based systems [3], testing is a fundamental lifecycle activity, with a huge economic impact if not performed adequately [16]. Such trends, combined with the needs for shorter delivery times and reduced costs, demand for the continuous improvement of software testing methods and tools, in order to make testing activities more effective and efficient.

Nowadays software is not more like simple applications but has evolved to large and complex system of systems [4]. A system of systems consists of a set

E. Cabello et al. (Eds.): ICSOFT 2016, CCIS 743, pp. 88–107, 2017.
DOI: 10.1007/978-3-319-62569-0_5

of small independent systems that together form a new system. The system of systems can be a combination of hardware components (sensors, mobile devices, servers, etc.) and software systems used to create big systems or ecosystems that can offer multiple different services. Currently, systems of systems capture a great interest from the software engineering research community. These type of systems are present in different domains like e-health [1] or transportation [17].

Testing these distributed and heterogeneous software systems or systems of systems, running over interconnected mobile and cloud-based platforms, is particularly important and challenging. Some of the challenges are: the difficulty to test the system as a whole due to the number and diversity of individual components; the difficulty to coordinate and synchronize the test participants and interactions, due to the distributed nature of the system; the difficulty to test the components individually, because of the dependencies on other components. Because of that, the attention from the research community increased, however, the issues addressed and solutions proposed have been primarily evaluated from the academic perspective, and not the viewpoint of the practitioner.

Hence, the main objective of this paper is to explore the viewpoint of practitioners with respect to the testing of distributed and heterogeneous systems (DHS), in order to assess the current state of the practice and identify opportunities and priorities for research and innovation initiatives. For that purpose, we conducted an exploratory survey that was responded by 147 software testing professionals that attended industry-oriented software testing conferences, and present the main results in this paper. Besides introductory questions for characterizing the respondents and contextualizing their responses, the survey contained several questions with the aim of assessing the practical relevance of testing DHS, the importance of testing several features of DHS, the current level of test automation and tool sourcing, and the desired features in test automation solutions for DHS. We expect that the results presented in the paper are of interest to researchers, tool vendors and service providers in the software testing field.

This paper extends our previous conference paper [11] with detailed explanations and new sections about background and test automation obstacles.

The rest of the paper is organized as follows: Sect. 2 presents some background on software testing concepts and terminology used in the survey. Section 3 presents the research method used to conduct the survey. Section 4 presents the results, which are further discussed in Sect. 5. Section 6 presents the conclusions of follow up interviews to further investigate drivers and barriers for DHS test automation. Section 7 describes the related work. Section 8 concludes the paper and points out future work.

2 Background

In this section it is presented some background on software testing concepts and terminology used in the survey.

2.1 Test Levels

There are different levels during the software testing process [2]. Typically the levels considered are: unit testing, integration testing, system testing and acceptance testing.

Unit testing is the testing of individual hardware or software units or groups of related units [7]. The goal of unit testing is to isolate each part of the program and show that individual parts are correct in terms of requirements and functionality.

In the case of a distributed and heterogeneous system comprising a set of interconnected components running on different machines or execution environments, unit testing usually refers to the testing of individual components. In automated testing, if a component under test calls other components, such components need to be simulated by test stubs (see definition of test stub in the next section).

Integration testing is the testing in which software and/or hardware components are combined and tested to evaluate the interaction between them [7].

In the case of integration testing of a distributed and heterogeneous system, besides checking the interactions of system components with the environment (users or external systems), it is also useful to check the interactions between components of the system, to improve fault detection and localization. Checking such interactions may be challenging, because of observability limitations.

System testing is the testing conducted on a complete, integrated system to evaluate the system's compliance with specified requirements [7]. System testing is concerned mainly with testing the interactions of the system with the environment (users or external systems), and evaluating extra-functional properties.

In the case of a distributed and heterogeneous system, interactions with the environment (users or external systems) typically occur at multiple locations. In automated testing, coordinating the test components that simulate those users or external systems is specially challenging, because of their distributed nature.

Acceptance testing is the formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable a customer, a user, or other authorized entity to determine whether or not to accept the system [7].

2.2 Test Harness

In software testing, a test harness is a collection of software and test data configured to test a program unit by running it under varying conditions and monitoring its behavior and outputs.

As shown in the Fig. 1, there are typically three main test components in a test harness: test driver, test stub and test monitor. The test drivers are responsible for calling the target code, simulating calling units or a user. In automatic testing they are also responsible for the implementation of test cases and procedures. The test stubs simulate modules, units or systems called by the target code; normally mock objects are used for this purpose. The test monitor is responsible to collect all the informations (or interactions) sent and received by the component under test. This information is important for fault diagnosis.

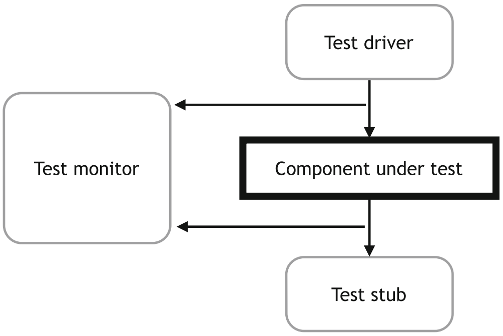

Fig. 1. Test harness.

2.3 Testing Methods

Software testing methods are traditionally divided into white-box testing [14], gray-box testing [12] and black-box testing [5].

In white-box testing, the internals of the system under test (SUT) are all visible. As a consequence, the knowledge about these internal matters can be used to create tests. Furthermore, white-box testing is not restricted to the detection of failures, but is also able to detect errors. Failures occur when there exist discrepancies between the specification and the behavior of the system. Errors are unexpected errors that occur while running the test. [13] Advantages are tests of higher quality because of the knowledge about system internals; however, there is also a big disadvantage, because looking into all aspects of a program requires a high effort.

In black-box testing, the SUT internal content is hidden from the tester. The tester only has knowledge about possible input and output values. The black-box testing only allows to test input-output functionality (functional testing). As an advantage, this technique is close to realistic conditions. One important disadvantage is the lack of internal information, which could be useful to generate tests, because sometimes it is necessary to know the content to test boundary values, that are normally responsible for failures.

In gray-box testing the advantages of both previous techniques are combined. The test design is realized at white-box level but the tests are executed at black-box level. For the tester, this has the advantage of having access to the SUT internal information while designing tests; however, the tests are executed under realistic conditions, where only failures are detected. Gray-box testing techniques are used for commercial model-based testing (MBT), where, e.g., the test model contains information about the internal structure of the SUT, but the SUT internal matters themselves are not accessible (e.g. for reasons of non-disclosure).

2.4 Test Automation

Several testing activities can be automated, with varying costs and benefits [15].

The testing activity that is most commonly automated is test execution. This requires that test cases are implemented as executable test scripts, test classes, etc. Support test components may also have to be developed, to act as test drivers, test monitors or test stubs. Automatic test execution is important to reduce the cost of regression testing, support iterative development approaches, enable load and performance testing, and avoid human errors that are common in manual test execution, among other reasons.

In the case of distributed and heterogeneous systems, automated test execution is specially challenging, because of the need to support multiple platforms and the need to coordinate the injection of test inputs and the monitoring of test outputs at distributed points, and very few testing frameworks exist for such systems [10].

Test coverage analysis is another testing activity that is commonly automated, usually in connection with automated test execution. Test coverage analysis is specially important in white-box testing, to determine parts of the code that are not being properly exercised, and help identifying additional test cases for increasing coverage.

Test case generation is usually performed manually. However, model-based testing (MBT) methods and tools have attracted increasing attention from industry, because of the ability to automatically generate test cases from system models. Based on behavioral models of the SUT, MBT tools are able to generate *abstract test cases*, which can be subsequently translated into *concrete test cases* ready for execution, based on mapping information between the model and the implementation.

In the case of distributed and heterogeneous systems, automated test case generation is specially challenging, because of the difficulty of modeling several characteristics inherent to such systems, such as timing aspects, concurrency aspects, and non-determinism, among other features, with very limited support provided by current MBT tools.

3 Research Method and Scope

The research method used in this work is the explanatory survey. Explanatory surveys aim at making explanatory claims about the population. For example, when studying how developers use a certain inspection technique [18].

3.1 Goal

The main goal of this work is to explore the testing of DHS from the point of view of industry practitioners, in order to assess the current state of the practice and identify opportunities and priorities for research and innovation initiatives.

More precisely, we aim at responding to the following research questions:

- **RQ1:** How relevant are DHS in the software testing practice?
- **RQ2:** What are the most important features to be tested in DHS?
- **RQ3:** What is the current status of test automation and tool sourcing for testing DHS?
- **RQ4:** What are the most desired features in test automation solutions for DHS?

3.2 Survey Distribution and Sampling

Since our main goal was to collect the point of view of industry practitioners that were involved in the testing of DHS, we shared the survey to the participants of two industry-oriented conferences in the software testing area: TESTING Portugal 2015[1] and User Conference on Advanced Automated Testing (UCAAT) 2015[2]. In total we distributed 250 surveys and we obtained 167 answers. From these 167 answers, only 147 were complete and valid. Most of the invalid answers were related with respondents that did not complete the survey.

3.3 Survey Organization

The survey was composed of two main parts. The first part was an introduction, where we explained the goal of the survey and define the term "Distributed and Heterogeneous Systems" in the context of this survey. In the context of this survey we define a Distributed and Heterogeneous System as a set of small independent systems that together form a new distributed system, combining hardware components and software systems, possibly involving mobile and cloud-based platforms.

The second part of the questions is divided in three different groups. The first group is related with the professional characterization of the participants. The second group contains questions about the company characterization. The last group contains the questions related with the testing of DHS and the main research questions underlying the survey.

4 Results

4.1 Participants Characterization

Before drawing conclusions on the main questions of this survey it is important to realize the profile of the survey participants. The results show that most of the people (70%) that responded this survey work in software testing, verification & validation and 41% are in the current position for more than five years (see Fig. 2).

Regarding the experience in software testing, the results show (see Fig. 3) that the majority of the survey participants have more than 5 years of experience in software testing in general and 40% have more than 5 years of experience with DHS.

4.2 Company Characterization

The companies surveyed worked in a large range of industry sectors. The results represented in Fig. 4 identify more than 10 different industry sectors.

[1] http://www.cvent.com/events/testing-portugal-2015/event-summary-a1a41d7f0867 4008b58e43454bb9f54a.aspx.
[2] http://www.etsi.org/news-events/events/868-2015-etsi-ucaat.

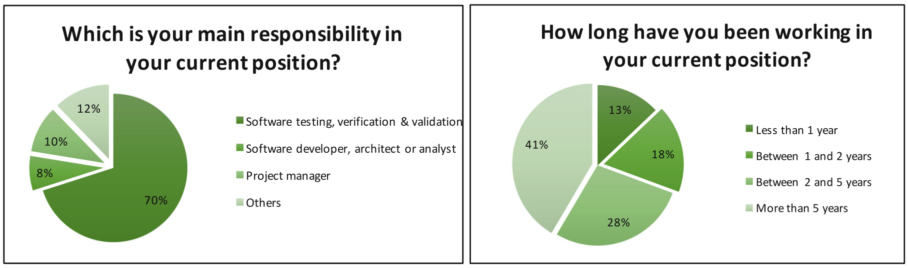

Fig. 2. Current position and time in current position [11].

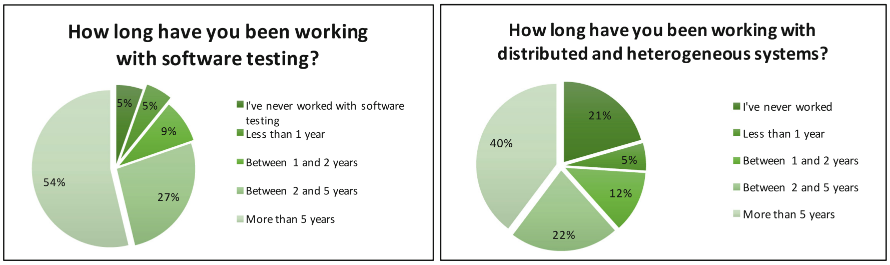

Fig. 3. Time in software testing [11].

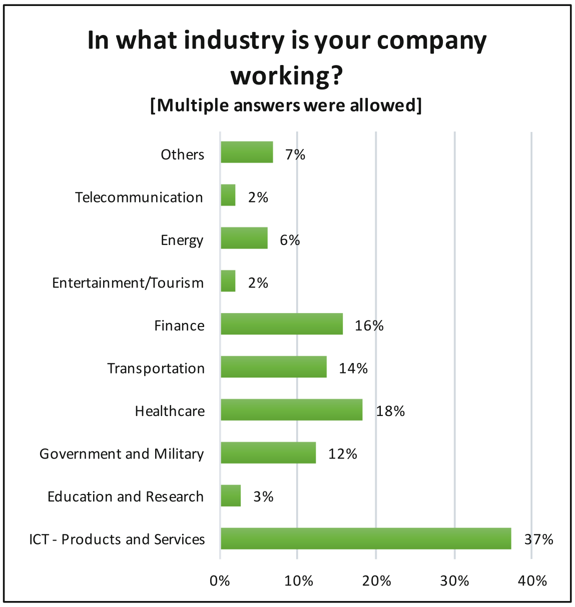

Fig. 4. Industry sectors [11].

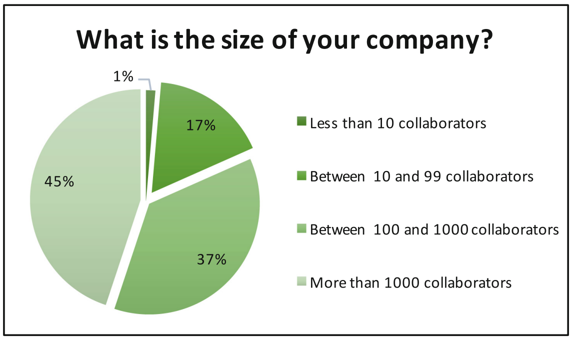

Fig. 5. Company size [11].

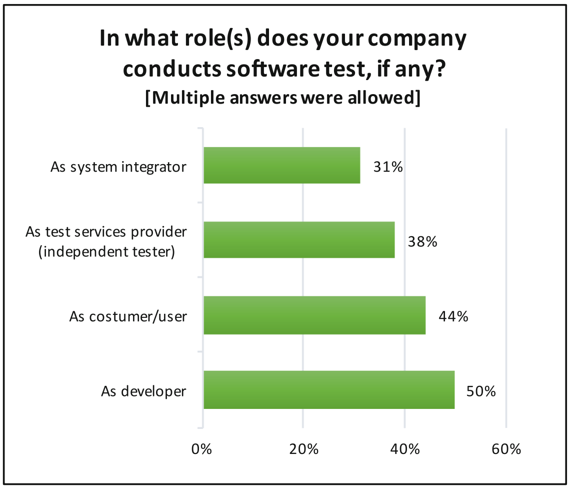

Fig. 6. Company roles [11].

We also analyzed the size of the companies according to their number of collaborators. Most of the companies are large companies, 37% have between 100 and 1,000 collaborators and 45% have more than 1,000 collaborators (Fig. 5).

The answers to 'In what role(s) does your company conducts software test, if any?' show that half of the companies performs tests to the software developed by themselves (Fig. 6).

Regarding the types of test levels performed, we realize from the answers (Fig. 7) that the unit testing level is the less performed and the other three levels (integration, system and acceptance) are performed with the same frequency.

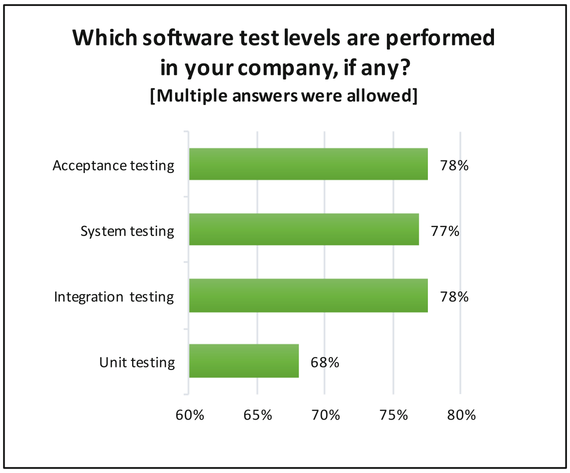

Fig. 7. Test levels [11].

4.3 Distributed and Heterogeneous Systems Testing

Focusing now on the main questions of this survey, specifically related to the testing of DHS, the answers to 'In what role(s) does your company conducts software test (for DHS), if any?' show that a vast majority of 90% of the companies (all but 10%) conducts tests for DHS in at least one role, with 42% of the companies performing tests for DHS developed by themselves (Fig. 8).

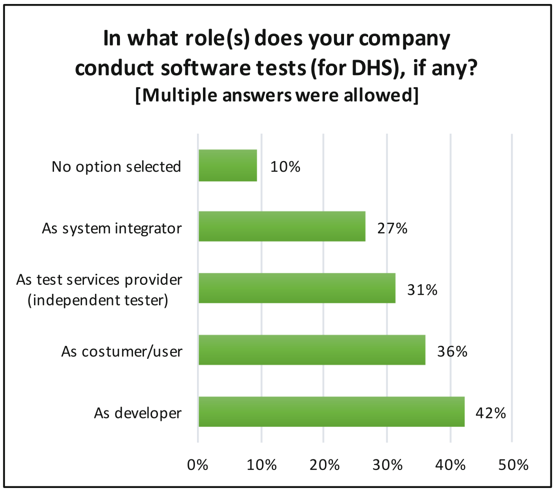

Fig. 8. Test roles DHS [11].

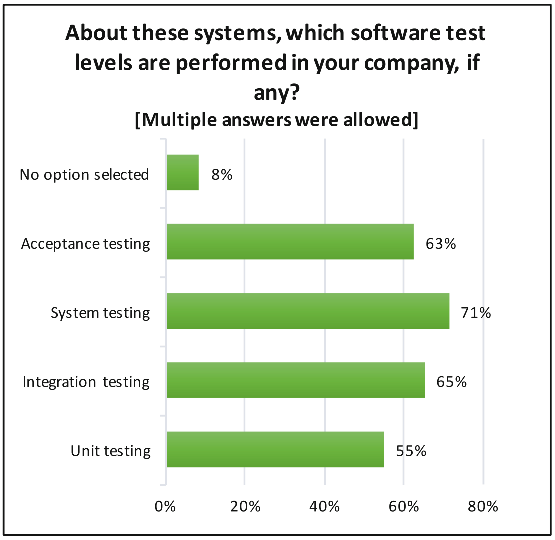

Fig. 9. Test levels DHS [11].

We also tried to understand what kinds of levels are most commonly used in the testing of such systems. Regarding the responses obtained (Fig. 9), there is a higher emphasis on system testing (71%) followed by integration testing (65%). Only 8% of the respondents did not mention any test level for DHS.

Regarding the most important features that need to be tested in DHS, the results in Fig. 10 show that the feature that was considered the most important to be tested was 'Interactions between components of the system' (with 76% of responses high or very high), followed by 'Interactions between the system and the environment' (71%) and 'Multiple platforms' (66%). All the features have been considered of 'very high' or 'high' importance by a majority of respondents (50% or more).

Regarding the level of test automation for DHS, the results presented in Fig. 11 show that 75% of the tests follow some automated process, however only 16% are fully automatic, which is lower than the 25% who claim to perform only manual testing.

For people who responded that there is at least some automatic process, we asked what kind of tool they use. With this question we can understand the level of effort required to automate the testing process. Looking at the results (Fig. 12) we realize that only 31% use a commercial tool to automate the process, and the majority, 69%, use a tool developed in-house, reusable or not in different SUTs.

Regarding the desired features of a test automation solution for DHS, the results presented in Fig. 13 show that the most important features (based in the percentage of responses high or very high) in an automated testing tool for DHS are 'Support for automatic test case execution' (75%) and 'Support for multiple platforms' (71%).

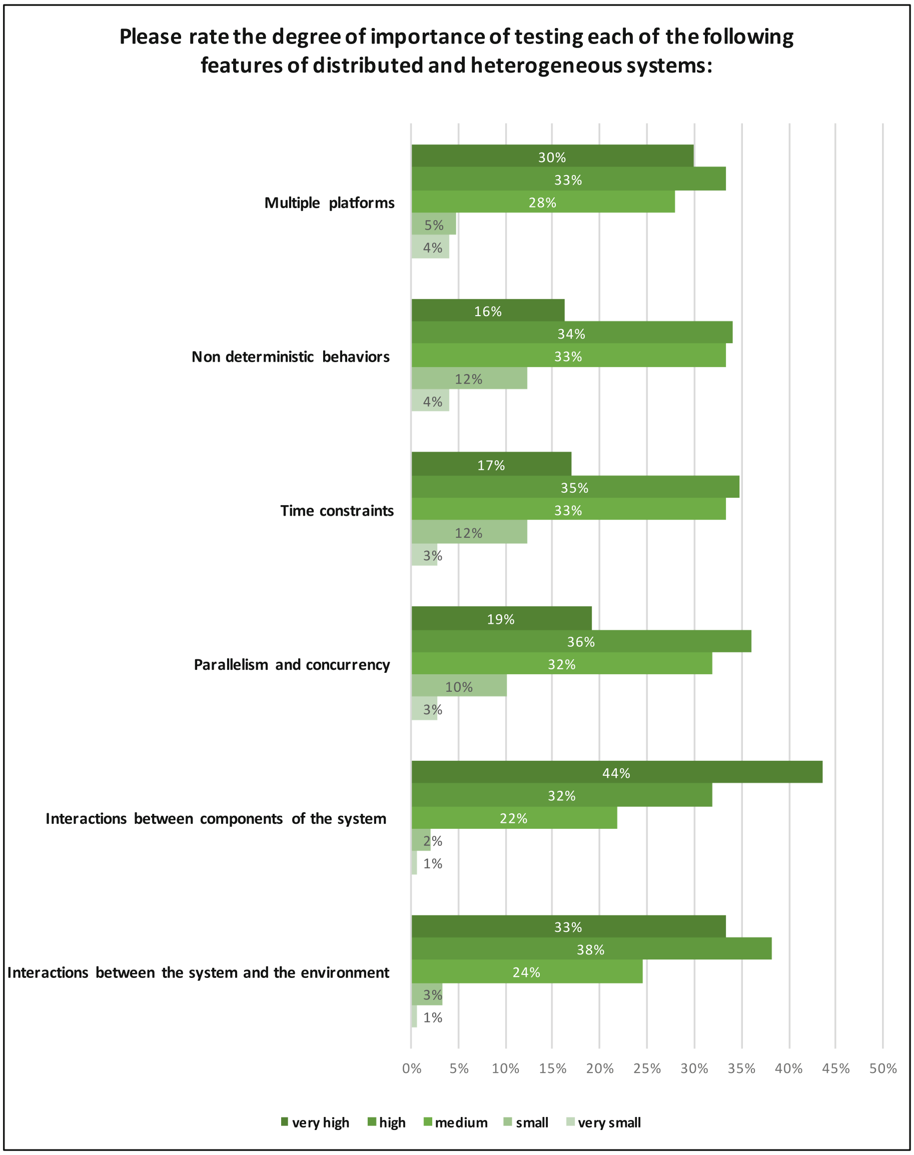

Fig. 10. Features.

As a possible solution to test DHS, we asked the participants in this survey if they would find useful a tool to test these systems that use only a model of interactions (UML sequence diagram) as an entry model. The results (Fig. 14) show that 86% consider useful a tool with these characteristics.

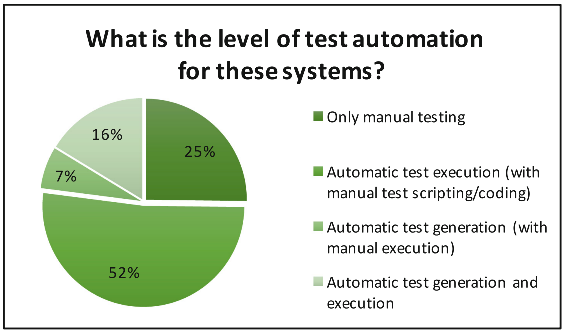

Fig. 11. Automation level [11].

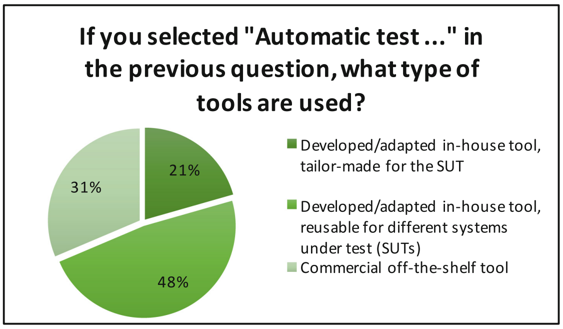

Fig. 12. Automation tool [11].

4.4 Multivariate Analysis

For questions specifically related to the opinion of the participants, a multivariate analysis was held with the aim to determine whether the participants' responses depend on their current function (Software testing, verification & validation versus all the others).

The results of the chi-square test for independence show that there is no statistically significant association (for a 95% significance level) between the current function (Software testing, verification & validation versus all others) and the answers to the questions shown in Figs. 10, 13 or 14.

5 Discussion

5.1 Relevance of Respondents

The results presented in the previous section show that this survey met the original purpose with regard to their target audience, since 70% of respondents' primary responsibility is related to 'Software testing, verification & validation'. With regard to their experience, the results showed that they are not only people

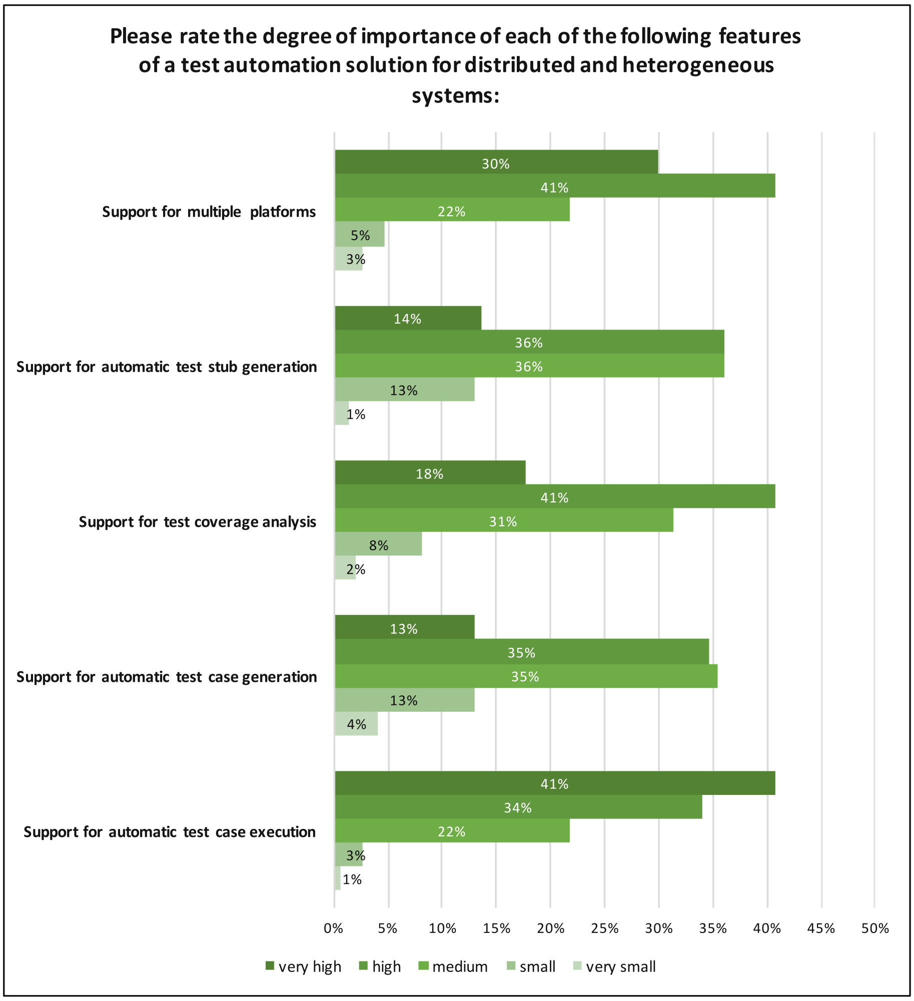

Fig. 13. Tool features.

who are mostly in their current position for several years, as work with software testing in general and specifically with DHS. With respect to the type of companies, the results show that this survey covers companies with diverse activity sectors and also large companies (45% have more than 1000 collaborators) which provides a great support to the conclusions reached.

Concerning the main conclusions we can draw from the results, they are next organized according to the initial research questions.

5.2 RQ1: How Relevant Are DHS in the Software Testing Practice?

The results (Fig. 8) show that a vast majority of approximately 90% of the companies surveyed (all with software testing activities in general) conducts

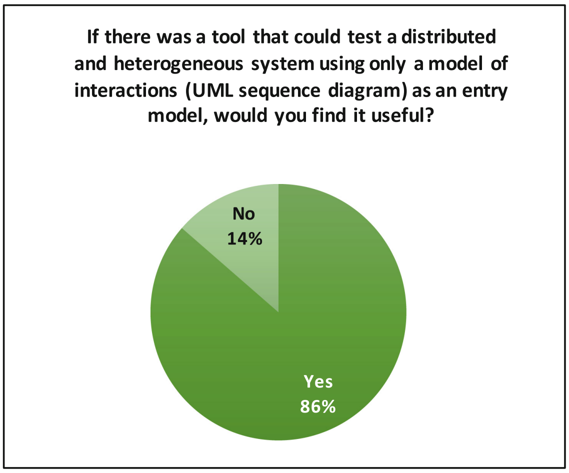

Fig. 14. New tool [11].

tests for DHS, in at least one role and at least one test level, hence confirming the high relevance of DHS in software testing practice.

5.3 RQ2: What Are the Most Important Features to Be Tested in DHS?

Regarding the most important features that need to be tested in DHS, the results in Fig. 10 show that the feature that was considered the most important to be tested was ‘Interactions between components of the system’ (with 76% of responses high or very high), followed by ‘Interactions between the system and the environment’ (71%) and ’Multiple platforms’ (66%).

Nevertheless, all the features inquired were considered of high or very high importance by a majority of respondents (50% or more).

5.4 RQ3: What Is the Current Status of Test Automation and Tool Sourcing for Testing DHS?

The results show that the current level of test automation for DHS is still very low, and there is large room for improvement, since 25% of companies in the survey claim that they only perform manual tests, against only 16% who claim to test DHS with a full automatic process.

If we look for companies that have some type of automation in its testing process, we realize that the automation process is requiring a high effort in the creation/adaptation of own tools, because only 31% of companies claim to use a commercial tool to test these types of systems.

5.5 RQ4: What Are the Most Desired Features in Test Automation Solutions for DHS?

Regarding the conclusions that can be drawn for future work, particularly at the level of creating tools that can reduce the effort required to test DHS, looking at Fig. 13, we realize that companies identify as key aspects of a tool to test such systems the ability to automate test execution (75% of responses with high or very high importancte) and the support for multiple platforms (71%).

Nevertheless, all the features inquired were considered of medium, high or very high importance by a large majority of respondents (83% or more).

The comparison of the degree of importance attributed to automatic test case execution (96% of the responses mentioning a medium, high or very high importance in Fig. 13) with the current status (78% of companies applying automatic text execution in Fig. 11), show that there is a significant gap yet to be filled between the current status and the desired status of automatic test case execution.

The gap is even bigger regarding automatic test case generation, with 83% of the responses mentioning a medium, high or very high importance in Fig. 13, and only 23% of the companies currently applying automatic text generation in Fig. 11.

We realized even by the Fig. 14, that companies are highly receptive to a test tool that has only a model of interactions as an input model for automatic test case generation and execution.

6 Case Based Analysis of Test Automation Obstacles

In order to analyze the drivers and barriers for DHS test automation in companies, we conducted follow-up interviews with some survey respondents. For the interviews we selected a sample of survey respondents from companies with different sizes (in terms of number of employees) and different test automation strategies.

6.1 Case A

Company A is a small company that develops software for external customers. The company performs manual tests on the software they develop. The justifications given by this company for only performing manual testing are:

- the low economic capacity to purchase commercial testing tools. Small companies and startups have limited budgets that have to be managed with many limitations and focusing on the rapid development of their products, leaving aside investments on test tools;
- the lack of human resources (in terms of availability and expertise) to allocate to test automation tasks. This type of company usually has a small number of employees, so all end up taking on various tasks. As the main focus is the rapid development of products, the use of any testing tool that requires additional learning time is immediately discarded;

- if they adopted a test automation solution, it would be costly to maintain automated test cases because of constant changes of product requirements and features, implying frequent changes in the test cases and test harness.

6.2 Case B

Company B is a large company that develops software for government and military areas. This company uses automation for test execution but still uses a manual process for test case generation (i.e., the creation of test scripts). For test automation, the company uses tools developed in-house, based on open-source frameworks. The justification given for not resorting to commercial tools is mainly due to the high costs charged by suppliers of these tools, often requiring the purchase of extra plug-ins for any additional feature needed.

Besides that reason, the following justifications were given for using tools developed in-house:

- to maintain knowledge within the company. Large companies prefer to develop their own tools because they are often unwilling to share information about their products with commercial tool vendors. For this reason they make use of open source tools that can be easily modified by the experts of the test area of the company itself;
- to be able to make adjustments to the tools more quickly, not being dependent on any vendor. Commercial tools leave companies dependent on their manufacturer, so large companies prefer to develop their own testing tools, since in the current market conditions it is increasingly important to have a quick reaction capability to modify the software.

With regard to manual test generation, the following justifications were given by the representative of the company for not using any automation process:

- the software developed by the company has very specific features that might be difficult to address with existing test generation tools;
- the lack of knowledge of company staff with regard to the creation of models needed as input for test case generation and the subsequent generation of test cases (this is one aspect that the company intends to improve in the near future).

6.3 Case C

Company C is a small to medium company that provides consulting services in the area of software testing. Most automation solutions that the company proposes and implements for its customers are related with test execution (and not test generation).

The reasons given for this have to do mainly with:

- the difficulty to automatically generate test cases;

- customers have no system model;
- the creation of system models requires a great effort;
- the system being tested is in a state of constant evolution and therefore not worth the effort in automatic test generating (or event automatic test execution).

As regards the tools that this company suggests to their customers for test execution automation, in most cases they recommend commercial testing tools. According to the representative of this company, this choice happens due to the following reasons:

- commercial tools are "ready to use";
- commercial tools do not require that the company has specialized human resources to adapt the test tool.

However, when the client has know-how in the test area, this company also indicates open source solutions that, in spite of requiring more maintenance, end up giving more flexibility and of course greater freedom to their users.

6.4 Synthesis

Analyzing the answers we have come to the conclusion that there are still several barriers and obstacles that prevent companies from adopting a fully automated test process.

Regarding the reasons for not adopting an automated test execution approach, we conclude that the main reasons are:

- cost of commercial testing tools (A);
- lack of human resources (availability and expertise) (A);
- frequent changes in the software under test (A, C).

For companies that have some level of automation in the testing tasks, the choice between commercial tools and in-house tools (usually based on open source tools) depends essentially on the type of company, since although the commercial tools are referred in the interviews as "ready to use" facilitating in this way the test automation process, they have several drawbacks, namely:

- are expensive, especially if extra functionalities and/or platforms are required (B);
- create too much dependence from vendors, and reduce flexibility for extensions and adaptations (B, C);
- know-how related with test automation is kept outside the company (B).

Regarding the reasons for choosing between a manual versus an automated test generation approach (with automatic test generation from models), we found:

- lack of human resources (availability and/or expertise) (A, B);
- frequent changes in the software under test (A, B, C);
- lack of system models (B, C);
- effort required for the creation of system models (B, C).

7 Related Work

We only found in literature one survey [6] that discuss some aspects related to the testing of heterogeneous systems. The survey conducted by [6] explored the testing of heterogeneous systems with respect to the usage and perceived usefulness of testing techniques used for heterogeneous systems from the point of view of industry practitioners in the context of practitioners involved in heterogeneous system development reporting their experience on heterogeneous system testing. For achieving this goal the authors tried to answer two research questions:

- RQ1: Which testing techniques are used to evaluate heterogeneous systems?
- RQ2: How do practitioners perceive the identified techniques with respect to a set of outcome variables?

The authors concluded that the most frequently used technique is exploratory manual testing, followed by combinatorial and search-based testing, and that the most positively perceived technique for testing heterogeneous systems was manual exploratory testing. Our work has a different objective of the survey conducted by Ghazi. The Ghazi main goal was to identify testing techniques, our aim is to understand how distributed systems and heterogeneous are tested in companies realizing which test levels are performed and which are the automation levels for testing these systems. The Ghazi survey also involved a much smaller number of participants (27).

As regards the general software testing in the literature there are many surveys, however as the main aim of our work is to analyze the state of practice, we analyze surveys carried out in the industry by recognized standardization bodies as ISTQB [8]. The most recent survey of this organization [9] conducted over more than 3,000 people from 89 countries, although it has a different purpose of our work because is related to the software test in general, provides results that meet the results presented in this article, namely that there are still significant improvement opportunities in test automation (was considered in this study the area with highest improvement potential).

8 Conclusions

In order to assess the current state of the practice regarding the testing of DHS and identify opportunities and priorities for research and innovation initiatives, we conducted an exploratory survey that was responded by 147 software testing professionals that attended industry-oriented software testing conferences.

The survey allowed us to confirm the high relevance of DHS in software testing practice, confirm and prioritize the relevance of testing features characteristics of DHS, confirm the existence of a significant gap between the current and the desired status of test automation for DHS, and confirm and prioritize the relevance of test automation features for DHS. The survey results indicated a limited adoption of complete test automation processes by companies.

For better understanding what are the obstacles that companies face for not adopting complete test automation approaches, we conducted follow-up interviews with companies of different sizes and testing approaches. The conclusions drawn from the interviews allowed us to identify some common obstacles, such as the cost of acquisition and difficulty of adaptation of test automation tools, the cost of test suite maintenance (namely with frequent changes in the software under test), and the effort and expertise required for the creation of system models needed as input for automatic test suite generation. We expect that the results presented in the paper are of interest to researchers, tool vendors and service providers in this field.

As future work, we intend to develop techniques and tools to support the automatic test generation and execution of test cases for DHS, addressing some of the obstacles previously identified, namely by using UML sequence diagrams as input for test generation to simplify the description of the SUT.

Acknowledgements. This research work was performed in scope of the project NanoSTIMA. Project "NanoSTIMA: Macro-to-Nano Human Sensing: Towards Integrated Multimodal Health Monitoring and Analytics/NORTE-01-0145-FEDER-000016" is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).

References

1. AAL4ALL: Ambient Assisted Living For All (2015). http://www.aal4all.org
2. Beizer, B.: Software Testing Techniques. Dreamtech Press (2003)
3. Boehm, B.: Some future software engineering opportunities and challenges. In: Nanz, S. (ed.) The Future of Software Engineering, pp. 1–32. Springer, Heidelberg (2011). doi:10.1007/978-3-642-15187-3_1
4. DoD: systems engineering guide for systems of systems. Technical report, Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering Version 1.0 (2008)
5. Edwards, S.H.: A framework for practical, automated black-box testing of component-based software. Softw. Test. Verification Reliab. **11**(2), 97–111 (2001)
6. Ghazi, A.N., Petersen, K., Börstler, J.: Heterogeneous systems testing techniques: an exploratory survey. In: Winkler, D., Biffl, S., Bergsmann, J. (eds.) SWQD 2015. LNBIP, vol. 200, pp. 67–85. Springer, Cham (2015). doi:10.1007/978-3-319-13251-8_5
7. IEEE: IEEE Standard Glossary of Software Engineering Terminology. IEEE Std 610.12-1990, pp. 1–84, December 1990
8. ISTQB: International Software Testing Qualifications Board, March 2016. http://www.istqb.org/
9. ISTQB: ISTQB worldwide software testing practices report 2015–2016. Technical report (2016). http://www.istqb.org/references/surveys/istqb-worldwide-software-testing-practices-report-2015-2016.html
10. Lima, B., Faria, J.P.: Automated testing of distributed and heterogeneous systems based on UML sequence diagrams. In: Lorenz, P., Cardoso, J., Maciaszek, L.A., Sinderen, M. (eds.) ICSOFT 2015. CCIS, vol. 586, pp. 380–396. Springer, Cham (2016). doi:10.1007/978-3-319-30142-6_21

11. Lima, B., Faria, J.P.: Testing distributed and heterogeneous systems: state of the practice. In: Proceedings of the 11th International Joint Conference on Software Technologies - Volume 1: ICSOFT-EA, pp. 69–78 (2016)
12. Linzhang, W., Jiesong, Y., Xiaofeng, Y., Jun, H., Xuandong, L., Guo, Z.: Generating test cases from UML activity diagram based on gray-box method. In: 11th Asia-Pacific Software Engineering Conference, pp. 284–291. IEEE (2004)
13. Mills, H.D., Dyer, M., Linger, R.C.: Cleanroom software engineering (1987)
14. Ostrand, T.: White-box testing. In: Encyclopedia of Software Engineering (2002)
15. Ramler, R., Wolfmaier, K.: Economic perspectives in test automation: balancing automated and manual testing with opportunity cost. In: Proceedings of the 2006 International Workshop on Automation of Software Test, AST 2006, NY, USA, pp. 85–91. ACM, New York (2006). http://doi.acm.org/10.1145/1138929.1138946
16. Tassey, G.: The Economic impacts of inadequate infrastructure for software testing. Technical report, National Institute of Standards and Technology (2002)
17. Torens, C., Ebrecht, L.: RemoteTest: a framework for testing distributed systems. In: 2010 Fifth International Conference on Software Engineering Advances (ICSEA), pp. 441–446, August 2010
18. Wohlin, C., Höst, M., Henningsson, K.: Empirical research methods in software engineering. In: Conradi, R., Wang, A.I. (eds.) Empirical Methods and Studies in Software Engineering, pp. 7–23. Springer, Heidelberg (2003)