Advance adherence to the upcoming standard ISO/IEC 29119[*]

Fernando R. Villas-Boas[†]
Sofist - Intelligent Software Testing
Campinas SP, Brazil
fernando.villasboas@sofist.com.br

Érika R. C. de Almeida
Institute of Computing, Unicamp
Campinas SP, Brazil
erikarca@gmail.com

Bruno T. de Abreu
Sofist - Intelligent Software Testing
Campinas SP, Brazil
bruno.abreu@sofist.com.br

Abstract

In this paper we study the upcoming new standard for software testing ISO/IEC 29119. The study stemmed from the need of conformance to standards for the automated software testing tool Crux, and it included its grounding standards and early drafts. We justify an early adherence to the standard and we show how this is being done in practice in the development of Crux.

1. Introduction

The Working Group 26 (WG26) of the ISO/IEC JTC1/SC7 Software and Systems Engineering committee has recently announced the release of advanced drafts of the new international software testing standard ISO/IEC 29119 Software Testing [6].

According to the working group,

> *The aim of ISO/IEC 29119 Software Testing is to provide one definitive standard that captures vocabulary, processes, documentation and techniques for the entire software testing lifecycle.*

This new standard will officially replace some existing IEEE and BSI standards for software testing, solving conflicts in definitions and procedures, and the announced timeline of the standard states that the final international standard will be published in June, 2012.

Whenever a new standard appears, particularly one announced to be published only in more than one year, some natural questions arise.

First, why select this standard as a main guideline for software testing in the place of others?

Second, assuming we are convinced about the upcoming standard but considering that it will be published only in more than one year, is there anything we should do about it at this early stage?

Finally, what can we do now in practice?

In this work we studied these two first questions and we show what we are doing in practice in the development of a tool for the automated generation of executable test cases.

The remainder of this work is organized as follows: Section 2 describes a brief history of the standards in software testing. Section 3 describes the outline of our proposal, while Section 4 presents our first conclusions and how we are going to continue our research.

2. History of standards in software testing

The first formal conference on software testing was held in June 1972 [3], and the first software testing book [3] was published as an outgrowth of the 1972 conference.

In the following years, national standards began developing and resulted in the IEEE 829 Standard for Software Test Documentation in 1983 [4] and IEEE 1008 Standard for Software Unit Testing in 1987 [5].

A natural critique of these standards followed.

In 1989, a meeting of the Specialist Interest Group on Software Testing (later to affiliate with the British Computer Society), agreed that

> *…existing testing standards are generally good standards within the scope which they cover, but they describe the importance of good test case selection, without being specific about how to choose and develop test cases.*

The group developed the standard BS-7925-2 Standard for Software Component Testing [1] and its companion

[*] The authors would like to thank the support of CNPq given through research grant #555473/2010-4.
[†] Corresponding author

standard for definitions, BS 7925-1 Vocabulary of Terms in Software Testing [2], published in 1998.

The group stated that

> *The most important attribute of this Standard is that it must be possible to say whether or not it has been followed in a particular case (i.e. it must be auditable). The Standard therefore also includes the concept of measuring testing which has been done for a component as well as the assessment of whether testing met defined targets.*
>
> *… the standard is deliberately limited in scope to cover only the lowest level of independently testable software. … the term "component" has been chosen rather than other common synonyms such as "unit", "module", or "program" to avoid confusion...*

Two points should be noticed from the group's comments. First, this is a standard intended to be auditable, thus having the necessary level of details and the necessary metrics for such auditing. Second, and probably because it was the first standard to address details of software testing, it is a standard deliberately limited in scope, i.e., unit testing.

Again, some natural critiques followed. In an early presentation in 2007, Murnane [8] pointed that

- *Existing standards do not cover all aspects of Software Testing Life Cycle:*
  - *BS 7925-2 only covers unit testing;*
  - *BS 7925-1 testing vocabulary written (exclusively) for BS 7925-2;*
  - *Missing higher level methods such as Use Case Testing and non-functional testing approaches such as Performance Testing, Security Testing, etc.*
  - *Risk-based testing and test strategy development not covered;*
- *Static Testing not covered;*
- *Potential conflict in definitions, processes and procedures;*
- *Practitioners may not know which standard to follow.*

These and other later critiques and motivation [9], such as the present lack in present standards of

a) organizational test policy and strategy,
b) project test management,
c) common system and acceptance testing techniques, and
d) non-functional testing,

as well as the sheer fact that the replacement of many standards into a single one is an important added value per se, led to the formation of a working group of the ISO/IEC JTC1/SC7 Software and Systems Engineering committee to address these issues, by means of a new standard, ISO/IEC 29119.

Its development began in May 2007, and currently comprises four parts: 1) Definitions and Vocabulary, 2) Test Process, 3) Test Documentation, and 4) Test Techniques.

It will officially replace the four IEEE and BSI standards for software testing mentioned before: IEEE 829 Test Documentation, IEEE 1008 Unit Testing, BS 7925-1 Vocabulary of Terms in Software Testing, and BS 7925-2 Software Component Testing Standard.

3. How to adhere in advance

Although the final publication of the standard is only due in 2012, there are many publicly available hints on how the standard will be, at least regarding its basis and outline.

First, the base standards for each part of the new standard are [9]:

- Part 1, Definitions & Vocabulary: *BS 7925-1*
- Part 2, Test Process: *BS 7925-2* and *IEEE 1008*
- Part 3, Test Documentation: *IEEE 829*
- Part 4, Test Techniques: *BS 7925-2*

Second, the final committee drafts for parts 2 and 3, and the committee drafts for parts 1 and 4 are already available, upon affiliation in the country's national standardization body (ABNT in Brazil) but subject to some nondisclosure clauses.

These drafts are not final, but they are mature enough to allow for the design of screens and forms, for instance, and to show in which direction we should plan future test processes.

Here we apply these ideas for advance adherence of an automated test tool and initially we stick to Test Documentation and Test Techniques.

3.1. Automated testing

Testing is a fundamental part of software quality evaluation, using metrics like the number of test cases in a test cycle, number of failures, and so on.

However, testing must have a well planned structure in order to guarantee a good coverage, as the lack of preparation, structure and steering can lead to time waste and retesting of the same functionalities, when more than one tester is involved.

Such coverage – and thus the final quality of the software – is best achieved when test activities are structured and follow accepted standards and good practices. These standards include BS-7925-1/2, IEEE-829 and IEEE-1008, which are being consolidated in the standard for software testing ISO/IEC 29119.

Many companies cannot fully afford to test, for lack of time and resources, and test automation has increa-

singly been used as a way to cope with this problem, since its initial cost pays off after a short time.

For this reason, Sofist, an outsourcing company specialized in software testing, is developing a tool named Crux for the automated generation of executable test cases. The tool considers the specification of the application and generates the corresponding test cases. As a result, test coverage is drastically increased and more faults can be identified before deployment, thus increasing the target software dependability.

The purpose of the tool is to ease testing activities. Besides producing automated test cases, it is also planned to follow the techniques and documentation recommendations proposed by ISO/IEC 29119.

The following sections show how we are developing the tool following these standards and how the quality of the test cases produced relates to the standards, even if presently only the grounding standards and early drafts are available.

3.2. Crux basic outline

Test automation is a complex and time consuming activity, as compared to the manual execution of tests, but after the scripts are created an automated test case can be executed repeatedly and in a faster way. This allows tests to be rapidly run after software changes, at the same time reducing the chances of human errors, common when tests are manually executed.

Since the creation of automated test cases is the most time consuming part in test automation, Sofist is developing Crux, a complete and versatile design tool for automated test cases for applications.

In the Crux environment, the system being tested is specified by defining its input interfaces and business rules. This allows the creation of sets of data, input actions and validation actions, which are combined in a test case and generate an automated test script. At this point scripts can already be run without intervention of the test analyst, as the tool also provides for the test data.

3.3. Pre-adherence to the standard

Since the new standard is yet to be published, in the design of the automated test tool we can only consider: a) the present standards upon which the new standard will be based and b) the drafts available.

Here we should mention a delicate issue.

Although the final committee drafts (FCD) for parts 2 and 3, and the committee drafts (CD) for parts 1 and 4 are available upon affiliation in the country's national standardization body, they are subject to nondisclosure clauses that hinder their presentation in this work. However, we did access them, and anyone can do the same, subject to affiliation in the country's national standardization body and to agreement with nondisclosure clauses.

Specifically in the case of Crux, a tool primarily aimed at software testers, many items of the new standard are still to be evaluated for actual use inside the tool. Presently we are focused only on parts 3 and 4 of the new standard, i.e., Test Documentation and Test Techniques. Furthermore, we are initially limiting adherence to test case specification in dynamic testing processes, and to testing techniques.

The final committee draft for test documentation in dynamic test processes is quite mature and resembles many parts of its base standard ISO/IEC 829.

Table 1 below shows what items of the standard (test documentation) are presently covered by Crux (marked with yes) and what items are now our to-do items that gradually will be included in the tool (marked with no).

Table 1. Test Documentation

Item	Covered by Crux?
Test case specification identifier	Yes
Test items	Yes
Input specifications	Yes
Output specifications	Yes
Environmental needs	No
Special procedural requirements	No
Intercase dependencies	Yes

As for testing techniques, exactly the same idea applies, but here we are initially relying only on BS 7925-2, as the final committee draft is not available yet. Table 2 shows present adherence (marked with yes) and to-do items that gradually will be included in the tool (marked with no).

Table 2. Testing Techniques

Technique	Covered by Crux
Equivalence Partitioning	Yes
Boundary Value Analysis	Yes
State Transition Testing	No
Cause-Effect Graphing	No
Syntax Testing	No
Statement Testing	No
Branch/Decision Testing	No
Data Flow Testing	No
Branch Condition Testing	No
Branch Condition Combination Testing	No
Modified Condition Decision Testing	No
LCSAJ Testing	No
Random Testing	No
Other Testing Techniques (Combinatorial Testing)	Yes

4. Conclusions

There are already some clear impacts that the new standard will have on the industry. The draft presently available for Part 2, Test Process, addresses the issue of conformance to the standard in a more profound way, deepening the effort toward auditability that began with BS 7925-2, but enlarging its scope to other test processes and for use during the complete software lifecycle. It explicitly describes the requirements for full conformance and for tailored conformance, which will allow external organizations to certify the conformance to the standard of a given organization's processes.

This is new in software testing and it meets the demand of organizations acquiring vital third party's software and of regulatory agencies. In a chain reaction, it will make software testing a more systematic and day to day part of software producers' life – and as part of this chain it will also affect outsourcing software testing companies like Sofist.

As a practical result, we decided to adopt a proactive attitude and adhere in advance to the standard.

Tables 1 and 2 in the previous section illustrate our approach for an advance adherence to the new standard, as a strategic decision. Many items are marked as no at this moment, meaning that now they are included in our plans for product development, and others marked as yes, meaning we already adhere to them.

However, the key point here is that now a steering standard in software testing is coming and it should be used for strategic planning.

As from the moment that WG26 announced the release of the working drafts of the new standard in July, 2010 [6], as well as the standards upon which it will be based and which it will officially replace, some points became clear.

First, ISO/IEC 29119 will probably be the most important standard for software testing, and for quite some time. The standard will be here to stay.

Second, 80% of the work of WG26 is complete, as the timeline of the standard elaboration is five years and it will be published in a little more than one year.

It is time to plan and prepare for it.

Finally, the following are some guidelines that we adopted and that might be useful for anyone involved in software testing.

- Obtain, study and focus on the base standards BS 7925-1/2, IEEE 829 and IEEE 1008;
- Get involved with the standard [7] and obtain the drafts already available. Beside allowing feedback for practitioners, the affiliation also allows access to the drafts;
- Start adhering now. Note which items are already complied with, and set a plan to adhere to the remaining items, as your company's strategic plan. We have a little more than one year left.

5. References

[1] BS 7925-2:1998. Software testing. Software component testing. BSI Group. London, 1998.

[2] BS-7925-1:1998. Software testing. Vocabulary. BSI Group. London, 1998.

[3] Hetzel, W.C. Ed. *Program Test Methods*. Prentice-Hall, Englewood Cliffs, N.J., 1973.

[4] IEEE 829-2008. IEEE Standard for Software and System Test Documentation. Institute of Electrical and Electronics Engineers. New York, (1983, 1998, 2008).

[5] IEEE 1008-1987. IEEE Standard for Software Unit Testing. Institute of Electrical and Electronics Engineers. New York, 1986.

[6] "ISO/IEC 29119 Software Testing." Internet: http://softwaretestingstandard.org/ accessed February 2, 2011. Working Group 26 (WG26) of the ISO/IEC JTC1/SC7 Software and Systems Engineering committee.

[7] "ISO/IEC 29119 Software Testing – How to get Involved". Internet: http://softwaretestingstandard.org/gettinginvolved.php updated August 03 2010, accessed February 2, 2011.

[8] Murnane, T. "ISO Standards on Testing". Test Automation Workshop 2007. Gold Coast, Australia, 2007. Internet: http://shakti.it.bond.edu.au/~sand/TAW07/ISO-SC7_TestStd29119.ppt accessed February 2, 2011.

[9] Presentation on ISO/IEC 29119 Software Testing. Internet: http://softwaretestingstandard.org/Downloads/ISO-IEC_29119_Software_Testing_July_2010.ppt accessed February 2, 2011.