December, 2009

testing experience

The Magazine for Professional Testers

Standards – What about it?

printed in Germany
print version 8,00 €
free digital version
www.testingexperience.com
ISSN 1866-5705

IEEE829:2008 A major change of focus

by Bernard Homès

The test plan is the basis for all future test activities on a software application. It is a contract between the test and development teams and the management.

IEEE 829:1998 was focused on documentation and this documentation detailed the strategy to be used for the testing activities for:

- Management to understand the added value of the tests and the remaining risks, the costs and requirements as well as the time frame,
- Development team to focus on the areas that are most critical or where a required level of quality has been set,
- Test team to develop and execute the test campaign on time and within budget,
- Test project manager to have a reference document to keep the project on course and on budget.

In 2008, the IEEE Standards Organization published a revised version of IEEE829, expanding it from software to also cover systems.

This article describes the different types of document used in the IEEE829:2008 standard and their relation to one another. Building on this information, it details the different steps and aspects of the test plan as described in the standard and the relations between the different chapters, how they complement each other in order to provide a complete solution

IEEE829:2008 introduces a number of changes, one of which is the change of focus from a document-centric standard to a process-centric standard.

This article maps changes from the old version with the new version, and suggests ways to implement the new version of the standard in your context.

Directly from one of the members of the IEEE829:2008 team, learn how to change your focus from documentation to testing tasks.

Note: This article was originally presented at CONQUEST09, but has since been improved

Expose

1 Key points

The following key points will be highlighted in this article:

1. Synthesis of the IEEE829:1998 and IEEE829:2008 standards
2. Relationship between the documents
3. How to be compliant to the 2008 version and how to migrate from 1998 to 2008 version
4. Advantages and drawbacks of the IEEE829:2008 version

1.1 Intended audience

The intended audience includes:

- Test and development managers as well as project managers and any testers, analysts or consultants intending to deploy the IEEE 829 standard.
- Other suggested audiences include business, line and purchase managers, in order to understand what is required in the context of testing their systems, so that they are able to evaluate proposed testing strategies.

2 Novelty

IEEE 829 is a well-known standard for software test documentation, that is not specifically new. The standard has been widely commented, sometimes attacked, by the different schools of testing, and is a bone of contention between the Agile testing school – which consider documentation to have very limited added value – and the systematic school, where the strict adherence to established processes and standards is considered mandatory.

Since 2003, a group of international experts from the different schools of testing have gathered under the auspices of the IEEE, in order to improve the standard and make it ready to face

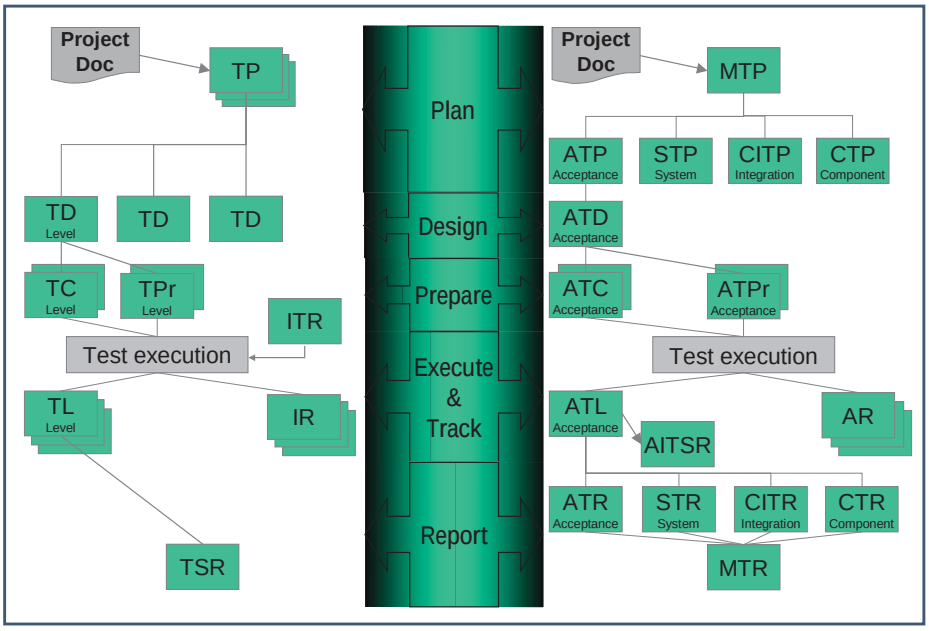

Figure 1: IEEE829:1998 vs IEEE829:2008 structure

the new challenges, adapting it to the evolved and more complex world facing us.

This endeavour reached its climax in July 2008 with the official publication of the IEEE829:2008 version of the standard, which now addresses both software and system test documentation. This expansion of scope from "software" to "systems" make this standard a leading item when one takes into account the increased complexity of systems and the emergence of systems-of-systems.

This paper describes the highlights of the new standard, its possible implementation in your or-ganization, and the tasks needed for you to become compliant with the new version of the standard.

2.1 Organization

The organization of the documentation suite of the new version of the standard, compared to the 1998 version is as described in the diagram (Figure 1). Shown on the left side is the IEEE829:1998 structure of documents, and on the right side the new 2008 version.

(See Figure 1)

Legend for IEEE829:1998 is as follows: TP: Test Plan; TD: Test Design; TC: Test Case; TPr: Test Procedure; ITR: Item Transmittal Report; TL: Test Log; TSR: Test Summary Report.

Legend for IEEE829:2008 is as follows: ATP: Acceptance Test Plan; STP: System Test Plan; CITP: Component Integration Test Plan; CTP: Component Test Plan; ATD: Acceptance Test Design; ATC: Acceptance Test Case; ATPr: Acceptance Test Procedure; ATL: Acceptance Test Log; AITR: Acceptance Interim Test Status Report; AR: Anomaly Report; ATR: Acceptance Test Report; STR: System Test Report; CITR: Component Integration Test Report; CTR: Component Test Report; MTR: Master Test Report.

3 Benefits

Compliance with a standard means that the person / organization intends to ensure that its processes and documentation fit a specific framework. This usually involves complying with mandatory (or regulatory) requirements and is based on the principle that an organization can improve its deliverables if the development processes are repeatable and measurable. In IEEE829:1998, the only items that were mandatory were the title of the documents and their chapters. This was sometimes considered as too burdensome and at other times too light-weight to fit the type of development undertaken.

With the advent of different SDLCs[1] and the increased complexity of some systems, it became clear that a new, updated and upgraded standard was necessary. This standard would be aligned to the ISO/IEC 12207 standard and adaptable to the criticality of the software and systems being developed.

1 SDLC : Software Design Life Cycle, described in ISO/IEC 12207

The inclusion of persons like James Bach and Cem Kaner from the Agile / context-driven school of testing ensured that the standard would focus on tasks that added value to the system and that it is adaptable to different contexts.

3.1 Applicability

Software becomes more and more pervasive nowadays. It is no longer limited to applications running on mainframes, minis or micros, linked or not by networks. It is now present in almost any commercial appliance, from fridges to cars and from telephones to personal assistants. The hardware verification aspect is now as important as the software-specific aspect. The new version of the IEEE829:2008 standard addresses this.

The new applicability of the standard is reflected in the decision to use the terms "software" and "system" interchangeably to reflect an "identified component or collection of components that include but are not restricted to software […] that comprise a software-based system".

This generalization to include hardware components implies that the future test managers and test analysts in charge of defining test strategies, test plans (Master Test Plans and Level Test Plans), will be required to have an understanding of the environment the software will be running on.

In terms of contexts, this standard can be applied in multiple contexts and environments, from aerospace to automotive industry, but also to other industries such as banking, insurance, and de-velopment of any software.

3.2 Tasks vs. documents

In IEEE829:1998, only a limited set of documents was suggested (see figure 1, left side), and testing tasks were defined based on the deliverables. In the new 2008 version of the IEEE829 standard, a number of tasks are provided per process, and each task has associated inputs and outputs, and can be associated with specific levels defined in ISO/IEC12207:2008.

This enables authors of test documentation to define the tasks required, and to use the standard as a checklist from which to pick and choose the different tasks according to their own context, ensuring that all tasks selected provide added value.

It is important to note the linkage between IEEE829:2008 and ISO/IEC12207. This allows mapping of testing tasks to other tasks not directly linked to systems development, such as acquisition.

As can be understood, this shift of focus from a document-oriented standard to an activity-focused standard enables the users to map their processes to the standard and check for any deficiencies. It is thus easier for users to implement this new version of the standard than the previous version.

Another aspect that should not be forgotten is the possibility to use this standard even when using agile methods and exploratory testing techniques that de-emphasize the use of documentation. As these methods and techniques are task-focused instead of being document-focused, compliance to the standard can be attained.

3.3 Integrity levels

Failure of some software and/or systems may lead to serious or even catastrophic consequences that cannot be solved by just rebooting the system.

Some examples include the Osprey V22 systems and the multiple developmental problems it suffered, killing in the process a number of test pilots. Other similar issues abound, so it was deemed important that the IEEE829:2008 standard addressed this in detail, showing what tasks were recommended at each level. These levels were called "integrity levels" and implement in IEEE829:2008 the ideas that were defined in DO178B/ED12B, ECSS and other standards.

The standard proposes a four-level integrity scheme, numbered 1 to 4 (4 is the highest integrity) described as follows:

- Negligible : "Software must execute correctly or intended function will not be realized causing negligible consequences. Mitigation not required."

- Marginal : "Software must execute correctly or an intended function will not be realized causing minor consequences. Complete mitigation possible."

- Critical : "Software must execute correctly or the intended use (mission) of system/software will not be realized causing serious consequences (permanent injury, major system degradation, environmental damage, economic or social impact). Partial-to-complete mitigation is possible."

- Catastrophic : "Software must execute correctly or grave consequences (loss of life, loss of system, environmental damage, economic or social loss) will occur. No mitigation is possible."

This concept of integrity level trickles down to other aspects of testing, such as the level of mandatory documentation according to the integrity level, the recommended minimum tasks per level and their intensity, and level of tester independence (see section 3.3.3). Integrity levels can also be defined based on the results of a risks analysis.

Even though the IEEE829:2008 standard does not mandate the use of integrity levels, it is noted that the use of an integrity level is "a recommended best practice that facilitates the selection of the most appropriate activities and tasks".

Incidentally, the implementation of integrity levels is also an incentive for defensive programming and other similar reliability enhancement techniques for software development.

3.3.1 *Mandatory documentation per integrity level*

The IEEE829:2008 standard proposes (table 2, page 16) a list of test documents associated to the integrity level of the systems or software being tested. An extract of this table (for integrity levels 4 and 1) is shown right:

A quick comparison of the two columns shows that for a software (or system) with "negligible" integrity level, the types of documents, number of documents and level of detail of the documents varies widely.

It is to be noted that this list of documents is not limitative, and that other documents may be needed in testing projects, such as risk analysis reports, etc. These other documents may also be tailored according to integrity level.

3.3.2 *Testing tasks & intensity per integrity level*

As with IEEE829:1998, the different testing tasks required are those needed to produce the docu-mentation, and any intermediate data that is needed. IEEE829:2008 proposes a table (table 3 pages 23-29) listing the tasks to be executed according to the integrity level. This table covers all processes specified in ISO/IEC12207:2008 (Acquisition, Supply, Development, Operation and Maintenance). An example is given below:

Integrity level 4 : Catastrophic	Integrity level 1 : Negligible
Master Test Plan	-- intentionally left blank, nothing mandatory --
Level Test Plan (Component, Component Integration, System, Acceptance)	Level Test Plan (Component Integration, System)
Level Test Design (Component, Component Integration, System, Acceptance)	Level Test Design (Component Integration, System)
Level Test Case (Component, Component Integration, System, Acceptance)	Level Test Case (Component Integration, System)
Level Test Procedure (Component, Component Integration, System, Acceptance)	Level Test Procedure (Component Integration, System)
Level Test Log (Component, Component Integration, System, Acceptance)	Level Test Log (Component Integration, System)
Anomaly Report	Anomaly Report (Component Integration, System)
Level Interim Test Status Report (Component, Component Integration, System, Acceptance)	-- intentionally left blank, nothing mandatory --
Level Test Report (Component, Component Integration, System, Acceptance)	Level Test Report (Component Integration, System)
Master Test Report	-- intentionally left blank, nothing mandatory --

Test Activities	Life Cycle Processes																																			
	Acquisition (5.2)				Supply (5.3)				Development (5.4)																				Operation (5.5)				Maintenance (5.6)			
	Acquisition Support Test (5.2.1)				Planning (5.3.1)				Concept (5.4.1)				Requirements (5.4.2)				Design (5.4.3)				Implementation (5.4.4)				Test (5.4.5)				Installation/Checkout (5.4.6)				Operation Test (5.5.1)				Maintenance Test (5.6.1)			
	Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level				Integrity Level			
	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1	4	3	2	1
System Test Plan generation													x	x	x	x																								
System Test Procedure generation																					x	x	x	x																
System Test Report generation																									x	x	x	x												
System test execution																									x	x	x	x												
Task iteration																																					x	x	x	x
Test Readiness Review participation																					x	x			x	x														
Master Test Report generation																													x	x	x	x					x	x		
Test Traceability Matrix generation									x	x	x		x	x	x		x	x	x		x	x	x		x	x	x													

Additional testing tasks that should be envisaged are also proposed by IEEE829:2008, per life cycle phase, and described in more detail in the standard (Annex D, pages 90-96). Among those additional tasks, we have [non-limitative list]: audits, branch coverage, database analysis, independent risk assessment, peer reviews, performance monitoring, regression analysis, certification, documentation evaluation, …

3.3.3 Tester independence

Tester independence is an important aspect that has been promoted in the ISTQB suite of syllabi, and which is a fact of life when studying the offshore testing markets.

The new IEEE829:2008 standard suggests a degree of independence between testers and developers according to the integrity level of the software / system being tested. This can be used as a guideline when decisions must be taken to outsource some testing activities.

Independence can be defined as (Te) Technical independence, (Ma) Managerial independence or (Fi) Financial independence. The test organizations proposed by IEEE829:2008 are :

- Embedded: where the tester is embedded in the development organization, such as a developer testing his/her own code
- Internal: where the tester is considered as inside the development team, but with a specific testing activity
- Integrated: where the testing organization is integrated in the same organization as de-velopment, but as a separate sub-organization
- Modified: where the testing organization is in the same overall organization, but not under the same direct management as the development team.
- Classical: where the testing organization is fully independent from the development or-ganization, either as a sub-contracted organization, or with the development being sub-contracted.

Alternative	Technical	Manage-ment	Financial
Classical	R	R	R
Modified	R	C	R
Integrated	C	R	R
Internal	C	C	C
Embedded	M	M	M

NOTE - R = Rigorous independence; C = Conditional independence; and M = Minimal independence.

The degree of independence varies between (R) Rigorous independence, (C) Conditional independence, and (M) Marginal independence as described in table F1 of IEEE829:2008.

One can compare this suggested level of independence with the one proposed by the ISTQB® in its Foundation Level syllabus. There are no major differences between the two.

In terms of the types of independence (Technical, Managerial or Financial), the reader will be able to evaluate the impact of each type of independence on the actual ratio of independence between developers and testers.

3.3.4 Test Level

Organization of the test documentation is always a problem. Should one group/segregate work according to the types of tests (static vs. dynamic), or according to the testing techniques (EQP, BVA, …), according to ISO9126 characteristics (functional, maintenance, etc), or according to the different testing levels (Component/Unit, Component Integration, System and Acceptance)? This new standard allows any type of organization, but it suggests one: Horizontal levels according to a standard SDLC.

IEEE829:2008 proposes to associate the documentation to the different test levels of the software / system being developed and tested. This implementation, which can be tailored to fit multiple integration levels, is detailed and ways to implement it according to the integrity level are also provided.

IEEE829:2008 suggests horizontal levels, going up from Component, to Component Integration, to System and up to Acceptance. This fits with the V-Model SDLC, and is fully in line with the ISTQB-proposed organization.

The Component test level, as well as the Component Integration test level have a different scope depending on the granularity of the component. If components are low-level, such as DLLs, the next level up is integration of the DLLs. If components are parts of a larger frame of reference, such as within a System-of-Systems, the components can be complete systems developed by suppliers, and the integration should be seen as "system integration" at the customer level.

Similarly, any one level suggested by the standard can be translated in more than one level for the organization, such as Acceptance being split in "Supplier System Acceptance" and "User Acceptance".

Your software and system documentation organization is fully customizable, and the standard can be adapted to your own context and organization.

The documentation suite for each test level comprises:

- The Level Test Plan (LTP) describing the scope of the test level, resources and methods
- The Level Test Design (LTD) providing more details/updates for the test methods
- The Level Test Case (LTC) with inputs and outputs, and
- The Level Test Procedures (LTPr) defining the test setups and execution instructions.

This implies that a full documentation suite will include Level Test Plans, Level Test Design documents, Level Test Cases and Level Test Procedures for all defined levels, resulting in a large documentation set.

IEEE829:2008 allows users of the standard to add, combine, or eliminate "whole documents and/or documentation content topics based on the needs (and integrity level) of their individual systems". This allows the test documentation to be fully tailored to the user context (see also section 3.3.6 below), which was not allowed in the previous (1998) version of the standard. This results in the user being allowed to provide reference to information present elsewhere, eliminate content topics provided in the process or covered by automation tools, combine or eliminate documents.

3.3.5 Customization & compliance

A number of organizations have already implemented IEEE829:1998 or some other type of docu-mentation suite. The adaptation of their existing documentation suite to fit the new standard should not be seen as too large a task, otherwise the changeover to the new standard will not occur.

Contrary to the 1998 version, the new version of the IEEE829 standard is highly adaptable, including in terms of the templates for the documents proposed in the standard. One will remember that IEEE829:1998 considered even the different sections of the documents mandatory. The 2008 version does not consider these sections to be mandatory. This does not mean that anyone can do anything and still be compliant to the standard. It means that the decision to depart from the suggested template must be justified and must be traceable.

As mentioned in a previous section, IEEE829:2008 allows the combination and/or elimination of documents, and suggests that this be done depending on the integrity level. The following table proposes a set of testing documents, depending on the Integrity Level:

The "L", representing "Level" in the titles of the documents should be replaced respectively with "A" for "Acceptance", "S" for "System", "CI" for "Component Integration" and "C" for "Component". The documents can be expanded, combined or eliminated depending on various factors such as duration of the development life-cycle, complexity of the system, higher level of integrity, number of test levels, outsourcing and geographically dispersed organization, personnel turnaround, etc.

Integrity Level	Selected documents
4	MTP, LTP, LTD, LTC, LTPr, LTL, AR, LITSR, LTR, MTR
3	MTP, LTP, LTD, LTC, LTPr, LTL, AR, LITSR, LTR, MTR
2	LTP, LTD, LTC, LTPr, LTR, LTL, AR, LITSR, LTR
1	LTP, LTD, LTC, LTPr, LTL, AR, LTR

IEEE829:2008 (chapter 6) proposes a number of ways to customize the standard to the user's needs.

Compliance to the standard can be claimed relatively easily, as the templates in the standard are not mandatory. It is thus relatively simple for the user to define a mapping between the user's documentation set and the templates. It is, however, necessary for the different topics to be "ad-dressed". IEEE829:2008 defines "addressed" as "*making a decision as to whether a documentation content topic is to be documented prior to the test execution (in a tool or not in a tool), documented post-test execution, not documented (addressed by the process), or not included. "Included" means that either the information is present or there is a reference to where it exists.*". It is thus necessary to document – for traceability and auditing purposes – the decision that led to a topic being "addressed" (or not) in the documentation.

Compliance to ISO/IEC12207 is attained by way of a traceability matrix describing the relationship between IEEE829:2008 and the ISO/IEC standard (Annex H of IEEE829).

3.3.6 Adaptability to context

The implementation of a standard in an organization should not entail major changes in the organ-ization's own processes. This standard provides suggestions on how to adapt the standard docu-mentation set to the organization's specific context. This enables the standard to be adaptable in terms of scope, depth and breadth.

The list of suggested testing tasks and processes, per integrity level, present in IEEE829:2008 enables compliance to be attained quickly, both in terms of process compliance and in terms of definition of the necessary tasks to attain compliance.

Process Compliance: In term of processes, the different processes are fully explained in the different sections of the standard, and these processes are fully in line with the ISO/IEC12207 software life cycle processes and with ISO/IEC15288 system life cycle processes. Standard processes in ISO/IEC12207 include : Management, Acquisition, Supply, Development, Operation and Maintenance.

- *Management processes*, where the MTP and LTP are generated and the test activities are monitored and evaluated continuously. Management processes include review activities, interfaces with organizational and supporting processes, and identification of process improvement activities.
- *Acquisition processes*, where the need to acquire a system, software product or service is defined, refined through the creation of a Request for Proposal (RfP) and the selection of a supplier. This process includes supplier management, from contract execution to acceptance, including testing of the supplied product or service (incl. associated do-cumentation).
- *Supply process*, from the decision to respond to a RfP, all the way to signing the contract, determining the required resources, preparation, execution and delivery of the software-based system. This process uses the contract requirement and overall schedules to revise and update the test interfaces planning between supplier and acquirer.
- *Development process*, consists of the activities and tasks of the development group, and includes analysis, design, code, component integration, testing, installation and acceptance related to the software or software-based product. It would be in this process that the applicable integrity level is evaluated and the traceability matrix generated.
- *Operations process*, covers the operation of the software product and operational support to the different users. This process performs operational testing, system operation, and user support. In this process we will also find risk identification and management (mitigation or avoidance), as well as identification of security issues.
- *Maintenance process*, which is related to any modification to the code or associated documentation, caused by identified problems, required improvements or adaptations. Usually any modification will be treated as a development process and verified as any of the other processes.

Tasks associated to processes: The tasks associated to the different testing processes are described in the standard (Annex C), including their required inputs and the produced outputs. It is thus easy to describe the required tasks and define the scope of activities that need to be executed (subcontracted, off-shored or done internally) and verified in terms of compliance to a service level agreement.

The user is thus guided in the implementation of the standard, with the necessary adaptation to integrity levels and to present (or perceived) risks.

Claiming compliance from IEEE829:1998 is possible by mapping the old (1998) version of the standard to the new (2008) version. This is relatively straightforward both in terms of number and scope of documents and in terms of content of the different sections of the documents. As the templates of ISO829:2008 are not mandatory (i.e. normative), compliance can be obtained relatively easily. Some differences appear in that the ITR (Item Transmittal Report) is not present in the new version of the standard, and in that there is a lack of "Approval" sections in the different documents.

Both of these can be implemented in your environment via customization of the standard.

Formal compliance can be claimed if all the topics in the standard have been "addressed" (see 3.3.5 above) and can be documented as having been covered.

3.4 Other changes

Beyond the changes from a document-centric to a process-centric standard, a number of more subtle changes have occurred in IEEE829:2008; some of these are listed below (non-exhaustive list):

- "Incident Report"is renamed to "Anomaly Report"
- "Item Transmittal Report" has been removed from the IEEE829:2008 documentation set. This does not mean that an item should be transmitted for testing any way you want, but that the formalism is no longer required
- "Level Interim Test Status Reports" are now available to provide interim reports on the test campaigns for each level. One is not limited to a single test report.
- Metrics are emphasized for Test Management purposes, and a specific "Master Test Plan Metrics Report" template is provided. Metrics can be selected from such standards as ISO/IEC9126-2:2003 External Metrics, ISO/IEC9126-3:2003 Internal Metrics and/or IEO/IEC9126-4:2004 Quality in Use Metrics.

In order to be complete, it is necessary to add that the IEEE829:2008 standard provides explanations for a number of key concepts that are emphasized in the standard:

- Integrity Levels (see also 3.3 above)
- Recommended minimum testing tasks for each integrity level (see also 3.3.2 above)
- Intensity and rigor applied to testing tasks
- Detailed criteria for testing tasks
- Systems viewpoint
- Selection of test documentation
- Compliance with International and IEEE standards.

4 Conclusion.

To the international testing community, the IEEE829:2008 is an important deliverable that is the result of a joint effort by different schools of testing to come up with a coherent and consistent set of documentation that can fit multiple contexts of software and system test.

As with any human endeavour, some points can still be improved. This article attempts to list the positive and less positive aspects and provides a number of pointers on how to avoid the negative points.

Mapping the old and new IEEE829 standards is straightforward and should not pose any problem to users who will be able to implement the new standard and convert existing documentation to fit the new IEEE829:2008 standard. This will enable the audience to ensure consistency of their do-cumentation across different projects, and across different versions of their software, also when outsourc-

ing either in NearShore or OffShore, and irrespective of whether working on simple systems or complex systems-of-systems.

The shift of focus from a document-oriented standard to a tasks-oriented standard, and its compliance with other standards, such as ISO/IEC12207:2008 and ISO/IEC15288:2008, ensure that the IEEE829:2008 standard will become as successful as its previous version.

References

[IEEE829:1998] IEEE Standard for Software Test Documentation, © IEEE, 1998

[IEEE829:2008] IEEE Standard for Software and System Test Documentation, © IEEE, 2008

[ISO12207:2008] Systems and software engineering — Software life cycle processes, © ISO/IEC, 2008

[ISO15288:2008] Systems and software engineering — System life cycle processes, © ISO/IEC, 2008

[DO178B/ED12B] Software Considerations in Airborne Systems and Equipment Certification, © RTCA/EUROCAE 2000

Biography

Founder and principal consultant for TESSCO Technologies Inc., Bernard Homès is a former member of the board of IEEE (France) and active in various software testing associations and workgroups.
After 25+ years in software development and testing with different consultancies, he set up the TESSCO group consultancies and provides training and consultancy services worldwide.
The fields of operation of TESSCO Technologies Inc. includes :

- Training testing teams & setting up software test centers,
- Specialization in mission-critical systems (banking, health care, telecoms, space & airborne systems),
- Qualification of Systems of Systems (large & complex critical systems) for international customers (such as Orange, Alcatel Alenia Space, Eurocopter).

Bernard Homès is also President of the French Software Testing Board (www.cftl.fr) and represents France at the ISTQB.
A long-time speaker at different international conferences and universities, Bernard has acquired a number of software testing certifications, and chairs the ISTQB Advanced Level syllabus working party.

He can be contacted at bhomes@tesscogroup.com