BS 7925–2: The Software Component Testing Standard

Stuart C. Reid
Cranfield University
Stuart.Reid@rmcs.cranfield.ac.uk

Abstract

This paper covers both the development of BS 7925-2 and compliance with the finished standard.

The process used to produce BS 7925-2 is described by presenting a history of its development, showing how initial chaotic development evolved with experience into a systematic process. This process was found to bear a strong resemblance to the software engineering process, and borrowed techniques from it, drawing particularly from those that support verification and validation. The knowledge gained from this development should be applicable to many areas of standardisation.

The most important clauses of BS 7925-2 describe a generic test process and definitions of test case design techniques and test measurement techniques. These are explained in detail and compliance with BS 7925-2 is described in the context of following the test process, which, in turn, requires the user to select techniques from the other two clauses.

1. Introduction

The software testing community has, until recently, lacked standard definitions of both test case design techniques and test coverage measures. This has led to the quality of testing performed by different organisations, or expected between customers and developers, to vary alarmingly. BS 7925-2 was produced to satisfy these shortcomings; it defines a generic test process, test case design techniques, and test coverage measures for use in component testing.

BS 7925-2 was conceived in early 1989 with the aim of being the first of a number of software testing standards developed by members of the British Computer Society Specialist Interest Group in Software Testing (SIGIST). After over eight years of development, copyright for this document, and an associated Vocabulary of software testing terms, was assigned to the British Standards Institution (BSI), allowing them to edit it (to conform to their documentation standards), and subsequently publish it in August 1998.

A component is defined for BS 7925-2 as a minimal program for which a separate specification is available. Component testing is often referred to as unit or module testing and is solely concerned with that dynamic testing at the very bottom of the 'V' life cycle model. Within this paper the term ‘component testing’ will be used throughout to maintain consistency with the terminology of BS 7925.

After this introduction, the process used to produce BS 7925-2 is described, including details of how the requirement for it was analysed and a brief history of its development. Next, suggestions on how to perform future standardisation work are presented, based on the experiences of the SIGIST working party (WP). The major contents of BS 7925-2 are presented and its generic test process is then used to demonstrate how users may comply with it. Finally, conclusions are provided.

2. Acronyms

ANSI	American National Standards Institute
BSI	British Standards Institution
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Organization for Standardization
MOD	(UK) Ministry of Defence
NWI	New Work Item
SIGIST	Specialist Interest Group in Software Testing

0-7695-0825-1/00 $10.00 © 2000 IEEE

SMARTIE	Standards and Methods Assessment using Rigorous Techniques in Industrial Environments
WP	Working Party
WWW	World Wide Web

3. The standardisation process

3.1. Conception

The development of BS 7925-2 was first proposed and taken up by the SIGIST in early 1989. It was recognised that no standards were available for software component testing and the stated initial aim was to produce a standard that defined how 'well' a software component was tested by dynamic testing. The intention was to start with component testing (as this task was considered to be of a manageable size) and then to subsequently produce further standards to gradually populate a framework of software testing standards.

3.2. Chaotic development

Work on BS 7925-2 continued for four years during which the informal working party (WP) convened at irregular intervals, normally immediately following SIGIST meetings. Development during this period was characterised by long periods between meetings (six months was not unknown) when small groups or individuals would work alone on areas they were interested in, and create sections (properly known as clauses in a standard) that were neither independently checked, nor planned as part of an integrated document. This inevitably led to difficulties in reaching a consensus on acceptance of the current working draft. Three drafts were issued in this initial period:

Issue 0.3 Jan. 1990 The initial issue for discussion within the SIGIST, formatted to BS0 [4], included test case design techniques and 'grades of testedness', but no test process.

Issue 1.2 Nov. 1990 The first 'public' version. Generally improved, and now including an initial definition of a test process. Guidelines were moved to appendices, which were expanded to include proformas for test documentation.

Issue 1.3 July 1992 Reformatted to European standards and incorporating numerous detailed technical changes as a result of comments received on Issue 1.2. The 'grades of testedness' were moved to the appendices.

3.3. Systematic development

As can be seen, the development up to this point was fairly unstructured with sections being inserted in one issue and being taken out in the next. As a result, in December 1992 it was agreed that a more systematic approach to the future development of BS 7925-2 was required and it was decided that a more formal WP, with appointed officers and a constitution would meet on a monthly basis.

At the first meeting of the new WP in January 1993 it was decided that an approach similar to the software development process should be used. In following an engineering approach to BS 7925-2's development, a requirements analysis was performed and a PERT chart was produced. The PERT chart, however, was only really useful in determining dependencies between activities as the enforcement of deadlines was not possible. Overall the application of traditional management techniques to the voluntary group work involved in producing BS 7925-2 was found to be impractical, and the prime means of motivating members of the WP to meet deadlines was the somewhat empty threat of passing the work on to someone else.

The first step in the requirements analysis was to determine whether there was still a requirement for BS 7925-2. This involved an investigation into the currently-available software testing standards, which concluded that there was still a requirement (specific details of this investigation are included in [29]). No work in progress by another group was discovered either, although when the author attended the Software Engineering Standards Symposium in 1993 he was informed about a new framework of software testing standards instigated by the IEEE. The current draft of BS 7925-2 was shown to representatives from the IEEE at the Symposium and they agreed that it should, when complete, form a part of this new framework. The requirements of BS 7925-2 were defined in terms of what it would, and what it would not provide to its users, and who they were. These documented requirements were found to be very useful in later discussions when deciding on the inclusion of new clauses.

A problem arose with the definitions produced prior to the formation of the new WP because they had not been widely reviewed and so no consensus had been reached. With hindsight it was decided that the drafts produced prior to 1993 would be considered as prototypes, which had been used to elicit requirements from reviewers, providing valuable functional and usability information. However, as is known in software development, the use of prototypes is fraught with difficulties. This was the case here, and discussion of material from pre-1993 wasted much time and resulted in many rewrites.

3.4. Internal reviews

As you would expect, the members of the WP had a strong software testing and quality assurance bias and both informal and formal review procedures were used. As each subsection was completed it was informally reviewed by the WP and any necessary revisions were carried out until a complete working draft was achieved. This typically required four iterations that, with the meetings being held monthly, led to a lengthy implementation and test phase before a working draft was completed.

In general, it was found that those techniques that were considered the most well-known and well-understood generated the most debate and therefore took the longest to define. This was probably because in these cases all the members of the WP had an opinion. In fact, a considerable proportion of this discussion was wasted due to a foreseeable problem that arose during the production of the guidelines clauses. It had been decided that the guidelines clauses of BS 7925-2 should include examples. For each test case design technique its application to an example component was provided, and for the test process the guidelines took the form of a set of example documentation describing the test process for an example project. Although the definitions had been reviewed and considered at length before production of the corresponding guidelines clauses began, in every case the definitions had to be changed due to feedback from this task, including the original definition of the component test process. In several cases discussion of the guidelines highlighted ambiguities in the corresponding definitions. Often, where a definition had eventually been agreed after much discussion, the example, by demonstrating a particular interpretation of this definition, re-started debate. In hindsight, the definitions and guidelines clauses should have been produced together as this would have avoided lengthy debates on the finer points of definitions that were later drastically changed due to feedback from producing the corresponding guidelines clauses.

Prior to publication of a new version of BS 7925-2, it was decided that the known benefits of formal software inspections should be applied. The inspection technique employed was that described in the book 'Software Inspection' by Tom Gilb and Dorothy Graham [14], and was led by Dorothy Graham herself, a member of the WP. The draft Standard was eventually released as Issue 2 in October 1994 at the EuroSTAR conference in Brussels.

3.5. External reviews

It was decided that the next draft of BS 7925-2 should be widely-publicised and made freely-available so that the widest audience of reviewers could be reached. It was publicised in the computing press and, to further widen the audience of external reviewers, a copy of it was published on the World Wide Web (WWW) in Spring 1996.

The most significant topic raised by external reviewers was that the current version of BS 7925-2 did not tell them how to determine whether they had complied with it. The WP had been so involved in defining the test case design techniques and measures (and their corresponding guidelines) that no-one had considered this fundamental point, most assuming that compliance had already been built-in to the structure of BS 7925-2. It was a fairly easy task to add the text to define compliance, but this did require the inclusion of clauses to allow users to define their own techniques, as long as they were defined in a similar manner to those already in BS 7925-2 (see 5.5 and 5.6).

3.6. Constraints on the inclusion of *new* material

The exclusion of syntax test coverage from BS 7925-2 and the use of Chow's switch coverage for state transition test measurement (detailed in [30]) highlight a problem the WP encountered throughout development. This was due to a self-imposed rule that only techniques that had been published elsewhere could appear in BS 7925-2. This rule was intended to stop the WP 'inventing' techniques, so enforcing a measure of consensus in BS 7925-2. This was a difficult rule to apply, especially when members of the WP had experience of applying 'better' versions of the techniques than those published. This was frustrating as the available published material was often quite old (for instance, Chow's switch coverage dates from a 1978 paper [7]) and published material on several techniques appeared to have stagnated since Myers' definitive work [24] of 1979. The solution was to add notes to the relevant guidelines clauses, as was done to highlight the benefits of the negative testing of state transition diagrams.

Small changes continued to be made to BS 7925-2 until April 1997 when it was finally handed-over to BSI. The one major exception to this was the inclusion of a new test case design technique - random testing. Initially there had been a reluctance to include random testing as no-one on the WP had much experience of using it and the published evidence of it being used successfully was not in a commercial environment. The author, however, presented results (for details see [31] and [32]) to the WP demonstrating its effectiveness, and based on this random testing was included.

3.7. Handover to BSI

From its very inception, it was intended that BS 7925-2 should become an international standard. The WP understood that this must be achieved by it first becoming a British Standard, via BSI. The first step on this route is to get the standard accepted by the relevant BSI working group (in this case IST/15) as a New Work Item (NWI). Each NWI needs a sponsor on the BSI working group, but for BS 7925-2 this presented no problem as Martyn Ould, who had first suggested BS 7925-2 in 1989, was a member of IST/15 and gladly took on this role. BS 7925-2 was accepted as a NWI in spring 1996 and the WP's final version was delivered to BSI in April 1997. This version was edited to the format required by BSO [4], and was then distributed as a Draft for Public Comment, to elicit comments, which had to be submitted by the end of December 1997. These comments were then considered by a technical panel, made up of members of the SIGIST WP and Martyn Ould, before a few final revisions were made and BS 7925-2 published as a British Standard in August 1998.

Once BSI decided to publish BS 7925-2, they approached the WP and requested that it be removed from the WWW and copyright assigned to them. This would then allow BSI to publish and sell it. The assignment of copyright caused some problems for the WP as several members used it in their work and wished to continue to be able to distribute copies of it both within their own organisations and to their clients. There was also reluctance to remove BS 7925-2 from the WWW as the WP felt that it should be as widely available as possible. BSI's position was that they had to make a profit, which they did by selling standards, and they could only do this if they owned the copyright (this would also protect them against illegal copying). The WP discussed this problem and eventually decided that the overriding goal should be to advance BS 7925-2 to the status of an international standard and therefore the demands of BSI should be accepted. At one point it was suggested that this aim could still be achieved if it was taken up by the American National Standards Institute (ANSI), via the IEEE, but as contact with the IEEE had been one-way for the previous couple of years (the WP sent them new versions, but heard little in return), it was thought best to remain with BSI. BSI stated that after BS 7925-2 was published, they would then propose it for fast-tracking by ISO to make it an international standard. By March 2000, enquiries revealed that BSI had forgotten to follow this through, although this was now progressing as a result of the author's query.

4. Future standardisation projects

The experience of producing BS 7925-2 has provided the following insights, or suggestions, for use in the development of new standards:

- Advertise your project on the WWW to reduce the risk of others duplicating your work, and inform your national standards body of your project as soon as you start.
- Set up a WP, comprising Chair, Deputy Chair, Secretary and named members, with rules to define membership and an agreed constitution. This formality provides for well-defined roles and a consistency of development that prevents decisions being overruled from meeting to meeting.
- Attempt to get a mix of members on the WP that are representative of the expected users of the standard.
- Perform research to ensure there is a requirement for the new standard.
- Produce a project plan, but be aware that traditional management techniques are often not applicable to managing groups of volunteers.
- Define objectives and analyse requirements, documenting both.
- Having agreed the requirements, define the structure of the proposed standard before populating it.
- Use template or exemplar clauses to show the style and format required of clauses.
- Produce the corresponding normative and informative clauses together, as they must be consistent and the necessary interpretation of the normative part to produce the informative part will often highlight potential problems.
- Define strict rules for introducing new clauses and reviewing existing clauses, as otherwise contentious decisions made at one meeting will be continually revisited at later meetings.
- Use a formal reviewing technique within the WP to review sections (and the complete standard) before making them available to external reviewers.
- Make draft versions of the standard available on the WWW so that as wide an audience as possible can review them.

- Use a standard for standards, such BS 0 [4], as a guide to the required final structure and terminology.

5. Compliance with BS 7925-2

The objective of BS 7925-2 is to enable users to directly improve the quality of their software testing, and so improve the quality of their products. As with most standards, BS 7925-2 does not mandate best practice as the scale of the move to best practice could act as a deterrent to its adoption for many users. Instead, BS 7925-2 mandates 'good practice' and requires users to document those attributes of their testing, such as the level of independence, which provide an indication of their test quality over and above this 'good practice'. Figure 1 shows the situation for an organisation which is currently non-compliant. Compliance with BS 7925-2 requires the non-compliant user to satisfy a number of requirements, which can be considered as steps, that should incrementally raise the level of testing quality until the user is fully compliant with BS 7925-2. Obviously some users will already satisfy the requirements of BS 7925-2, but will still benefit from the recognition of the quality of their testing reflected by their compliance with it.

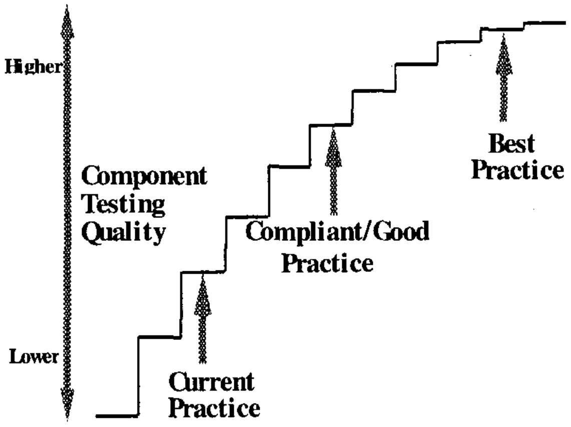

Figure 1. Testing quality and practice

5.1. The content of BS 7925-2

BS 7925-2 comprises two parts; the normative part, which defines those requirements that *must* be satisfied to comply with BS 7925-2, and the informative annexes, which provide support to the normative part by providing examples and advice. The normative part comprises six clauses:

1. Scope
2. Normative References
3. Definitions
4. Process
5. Test Case Design Techniques
6. Test Measurement Techniques

5.2. Scope

This clause defines the objectives of BS 7925-2, stating that it specifies a process for dynamic component testing and techniques for the design and measurement of that testing. This clause also defines the extent and limitations of BS 7925-2. As such, it excludes static testing and testing in other phases of the life cycle, such as integration, system and acceptance testing. It also explains that the functional (black box) techniques apply to components written in any language, whereas the structural (white box) techniques are restricted to the testing of components written in procedural languages.

5.3. Normative references and definitions

The 'normative references' clause lists other standards referenced by BS 7925-2 and includes only BS 7925-1, the Vocabulary of testing terms.

All definitions used in BS 7925-2 are presented in BS 7925-1, the Vocabulary of testing terms. Initially BS 7925-2 contained a large definitions section as no existing standard could be found that provided a suitable range of definitions. Because BS 7925-2 is intended to be the first of a number of software testing standards it was decided that a single standard of software testing terminology should be produced to support this framework. The definitions section of BS 7925-2 was therefore removed to form the basis for this separate vocabulary standard. BS 7925-1 was produced and reviewed in parallel with BS 7925-2 because the latter relies on the availability of the definitions in the former.

5.4. Process

Clause 4 specifies the generic test process to be followed for software component testing. It mandates that at the project level there must be a project component test strategy and a project component test plan. It also mandates that each component to be tested must have a specification from which it is possible to derive the expected outcome for a given set of inputs.

The project component test strategy requires the specification of:

- the test case design techniques, which must be selected from clause 5 (see 5.5);
- the criterion for test completion (including rationale for its selection), which must be measured using techniques chosen from clause 6 (see 5.6);
- the degree of independence of the testers;
- the approach to component testing (e.g. isolation, top-down, bottom-up, etc.);
- the test environment, including both hardware and software requirements;
- the test process to be used for software component testing - the generic component test process from BS 7925-2 is illustrated in Figure 2.

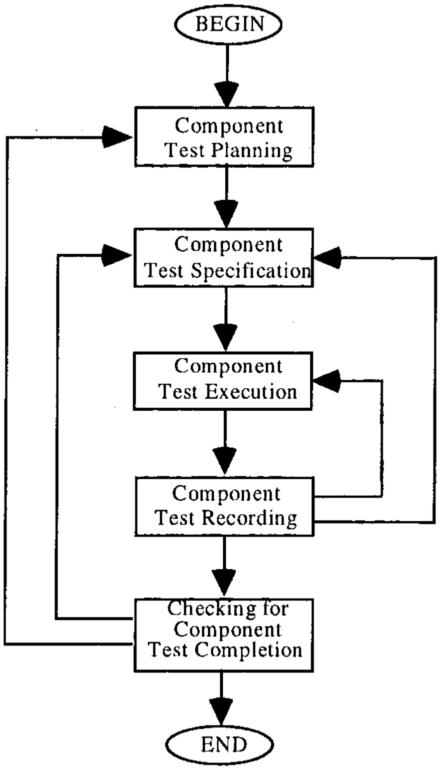

Figure 2. Generic component test process

The project component test plan requires the specification of dependencies between component tests, and their sequence.

The sequence of activities defined by the generic test process must be followed for each test case. More specific requirements for each of the individual activities of the process are described in the following subsections.

5.4.1. Component test planning

- Specify how the project component test strategy and the project component test plan apply to this component, listing any exceptions.

5.4.2. Component test specification

- Specify test inputs for each test, selected using a test case design technique from clause 5 (see 5.5).
- Specify the expected outcomes for each test - in advance.
- Document the point of each test.
- All test cases shall be repeatable.

5.4.3. Component test execution

- Each test case in the test specification shall be run.

5.4.4. Component test recording

- Check and record the results of each test.
- Analyse the reasons for failure to subsequently allow the removal of the fault.
- Record the test coverage achieved (in terms of the test completion criterion).
- Record the test configuration.

5.4.5. Checking for component test completion

- Check whether the test completion criterion is satisfied.

5.5. Test case design techniques

In the project component test strategy, the choice of test case design technique(s) to be used to generate the test inputs in the Component Test Specification activity (5.4.2) must be documented. It is mandated that this choice must be made from the test case design techniques defined in clause 5, which are listed below:

- Equivalence Partitioning
- Boundary Value Analysis
- State Transition Testing
- Cause-Effect Graphing
- Syntax Testing
- Statement Testing
- Branch/Decision Testing
- Data Flow Testing
- Branch Condition Testing

- Branch Condition Combination Testing
- Modified Condition Decision Testing
- LCSAJ (Linear Code Sequence And Jump) Testing
- Random Testing
- Other Testing Techniques

It can be seen that, with the final entry, this is an exhaustive list of all available test case design techniques as it allows the use of techniques not specifically defined in the clause. This mechanism has been included to allow new techniques to be used and still permit users to comply with BS 7925-2. Such techniques can be used as long as the technique is both available in the public domain and documented in the same manner as the other techniques included in BS 7925-2.

BS 7925-2 only provides guidance on the choice of which test case design techniques and test completion criteria to use. The main reason for this is that there is no established consensus on which techniques and criteria are the most effective. The selection will vary as it should be dependent on a number of factors such as criticality, application area, and cost, among others. Research into the relative effectiveness of test case design and measurement techniques has, so far, produced no definitive results.

5.6. Test measurement techniques

Test measurement techniques are handled in a similar manner to the test case design techniques. The project component test strategy must include the choice of test measurement technique(s) to be used as a test completion criterion in the Component Test Recording activity (5.4.4). This choice must be made from the test measurement techniques defined in clause 6, which are listed below:

- Equivalence Partition Coverage
- Boundary Value Analysis Coverage
- State Transition Coverage
- Cause-Effect Coverage
- Statement Coverage
- Branch/Decision Coverage
- Data Flow Coverage
- Branch Condition Coverage
- Branch Condition Combination Coverage
- Modified Condition Decision Coverage
- LCSAJ (Linear Code Sequence And Jump) Coverage
- Other Test Measurement Techniques

Again, the same mechanism as described in 5.5 has been included to allow new techniques to be used and still permit users to comply with BS 7925-2.

There is not a direct correspondence between the test case design techniques and test measurement techniques (as there are no definitions of syntax test coverage and random test coverage). Therefore, it is not necessary to choose the same test measurement technique and test case design technique. In fact, the practice of using functional test case design techniques and structural test measurement techniques is recommended. Functional techniques are effective at detecting errors of omission, while structural techniques can only detect errors of commission. So a test plan could typically require boundary value analysis to be used to generate an initial set of test cases, while also requiring 100% branch coverage to be achieved. This diverse approach would, presumably, lead to branch testing being used to generate the supplementary test cases required to achieve coverage of any branches missed by the boundary value analysis test case suite.

BS 7925-2 recommends that the test coverage levels chosen as test completion criteria should, wherever possible, be 100%. Strict definitions of test coverage levels have sometimes made this level of coverage impracticable, however the definitions in clause 6 of BS 7925-2 have been defined to allow infeasible coverage items to be discounted from the calculations thus making 100% coverage an achievable goal.

6. The relationship with other standards

It has long been the intention that BS 7925-2 would align with the ISO 9000 series of quality assurance standards, which require software developers to use software testing standards. It is expected that BS 7925-2 would be called from within a quality management system compliant with ISO 9001.

A number of standards specify that particular test case design techniques and measures are to be used, but they rarely provide definitions of these techniques, apparently assuming that they are well-understood and not open to interpretation. The experience of the WP has been that these well-understood techniques have taken the most effort to agree on and define. For instance, a problem with defining branch testing and coverage, as described by Bruce Elliott [11], fuelled discussions over a two year period and this was not an isolated case. The provision of definitions of test case design techniques and measures by BS 7925-2 should now prevent any future disagreements over interpretation occurring.

A number of existing standards covered software component testing to different degrees when work on BS 7925-2 first began, but it was felt that there was no useful software component testing standard. As stated earlier, one of the first tasks of the WP was to determine that there was still a real requirement for BS 7925-2 and to ensure that it did not duplicate material already available. Among others, the following standards were considered: [1], [2], [3], [6], [8], [9], [17], [18], [19], [27] and [28]. The result of this research confirmed the original belief that there was still a requirement for BS 7925-2. This position was supported by Wichmann, who stated that the use of ISO-9001 for software development would encourage a move towards software test measurement and its standardisation [36] and specifically suggested that BS 7925-2 could fulfil the role identified [35].

Estimates of the number of software-related standards vary from "more than 300" [34] to "over 1100" [25], but there is no disagreement that within software standards there is duplication, inconsistencies, overlap and omissions. In 1987 IEC and ISO formed a joint technical committee to attempt to impose some structure in this area. The result was the JTC1/SC7 Architecture [34]. One aim of this architecture is to position software engineering standards in relation to quality systems, concentrating on the ISO 9000 series [22]; BS 7925-2 was produced with the expectation that it would be used to specify the software testing requirements of an ISO 9000 quality system. The JTC1/SC7 architecture is based on the generic process from the software life cycle processes standard [20]. At the next level, users will require standards to support particular phases of the life cycle, such as testing. There are a number of standards that concentrate solely on testing including [1], [2], [5], [15], [16] and [21]. As stated earlier, however, none of these covers component testing satisfactorily, leading to the requirement for BS 7925-2.

An orthogonal set of software standards that include software testing requirements are the application-specific standards, such as [9], [10], [15], [17], [23], [26] and [33]. These standards define software requirements for particular areas, such as avionics or railway signalling systems, and specify component testing requirements, but without defining the specified techniques. Any updates to such standards or new application-specific standards should specify and component testing requirements by referencing BS 7925-2.

As part of the four year SMARTIE project, Fenton [12] reported on a measurement-based approach to assessing software engineering standards. He stated that it was impossible to make an objective assessment of conformance to the majority of existing standards and went on to make recommendations on the future writing of standards. When reporting on software testing standards [13] he stated that BS 7925-2 was "as close to a true engineering standard as any seen by SMARTIE".

7. Conclusions

This paper initially described the development of BS 7925-2, the standard for software component testing, and provided suggestions for future standardisation efforts based on that experience. It then went on to describe the activities necessary for users to comply with BS 7925-2.

BS 7925-2 took a long time to produce and during the first four years of development, before a systematic approach was taken, much effort was wasted. Only after the adoption of a structured development process, similar to that used in software development, was real progress made. The move from chaotic development to a more defined process was difficult, but hopefully the experience of the WP will not be lost and the lessons learnt applied to any future projects undertaken. It is assumed that those working under the auspices of BSI are given more guidance in this respect, but there certainly remains a requirement for more information in this area, and, as the national standards body, this should be BSI's responsibility. The knowledge gained from the experience of the standardisation process described here will be of use to anyone producing a standard, but especially to those who intend taking a standard to national status, such as through BSI.

It is the author's belief that in the near future new technology has a major role to play in the standardisation process. The WP meetings were held monthly in London and for most of those attending much of the day was spent travelling. Although meetings are still necessary, when videoconferencing becomes more widely available then virtual meetings may be held using this facility. This will allow the WP to include a wider range of members, as participants may then be located anywhere in the World, not just within travelling distance of London. The WP found over the lifetime of the development that the increasing availability of quick and cheap communication using email gradually made the dissemination of minutes and changes for review far easier. The WWW also allows standards to be reviewed by a potentially vast audience, thereby making them true consensus standards.

The WWW can affect a more fundamental change to the standards arena. Standards should serve the common good, and so the more widely-available they are, the better. The WWW provides an opportunity to make standards much more widely accessible, but, so far, standards organisations have prevented its use in this way as they will thereby lose a source of income. Although these bodies have a valid argument for selling standards whose production they have funded, it seems unreasonable that a monopoly (the national body alone can put forward standards to ISO) should be allowed to charge for standards produced by others at no cost to themselves. The author hopes that BS 7925-2 will some day return to being freely available on the WWW, along with other standards.

The framework of software testing standards instigated by the IEEE as part of their Master Plan for Software Engineering Standards seems to have disappeared. When work on BS 7925-2 started the intention was that it would become part of this framework. If the IEEE initiative has failed then a new one needs to be set up. A new framework would be expected to accommodate standards on integration, system and acceptance testing, as well as application-specific testing standards, such as user interface testing, client-server testing, internet/intranet testing, etc. The SIGIST WP, now it has finished BS 7925-2, started work on a standard on techniques for non-functional testing in September 1998. This has allowed the expertise built up in the WP, most especially in the areas of procedure, to be maintained.

BS 7925-2, now published by BSI, should fill a gap in the existing software engineering standards framework, and will hopefully go on to become an international standard. As with all standards it is not perfect - any product of the standardisation process, which is driven by the need for consensus, cannot be perfect - but in comparison to the majority of standards it is of high quality. It is already being used commercially by several major organisations, including government agencies, the banking sector, software houses, etc. The UK MOD also use it to specify their component testing requirements. At present many application-specific standards mandate that component testing is performed but rarely define the requirement clearly. A study should be performed to identify where component testing is specified in other standards and the authors of these standards could then be made aware of BS 7925-2 so that their documents may refer to it, rather than attempt to re-define the required component testing themselves.

8. References

[1] ANSI/IEEE Std 1008, *Standard for Software Unit Testing*, 1987.

[2] ANSI/IEEE Std 1012, *Standard for Software Verification and Validation Plans*, 1986.

[3] ANSI/IEEE Std 829, *Standard for Software Test Documentation*, 1983.

[4] BS 0, *A standard for standards*, BSI, 1991.

[5] BS 5887, *Codes of practice for Testing of Computer-Based Systems*, 1980.

[6] BS 7165, *Recommendations for the Achievement of Quality in Software.*

[7] Chow, T.S. 'Testing Software Design Modelled by Finite State Machines' *IEEE Trans. on S/W Eng.*, Vol.SE-4, No.3, May 1978.

[8] Def Stan 00-17, *Modular Approach to Software Construction, Operation and Test (MASCOT)*, Issue 1.

[9] Def Stan 00-55, *The Procurement of Safety-Critical Software in Defence Equipment*, Interim, 1991.

[10] *Development Guidelines for Vehicle-Based Software*, MISRA, 1994.

[11] Elliott, B. 'A Standard for Software Component Testing', *Proc. of the 2nd European Conference on Software Testing, Analysis and Review*, Brussels, October 1994.

[12] Fenton, N. and Page, S., 'Towards the Evaluation of Software Engineering Standards', *Proc. of the Software Engineering Standards Symposium*, Brighton, UK, 1993. IEEE Computer Society Press, 1993.

[13] Fenton, N., 'Measurement and Testing Standards', *Presentation to BCS SIGIST*, London, May 1993.

[14] Gilb, T. and Graham, D., *Software Inspection*, Addison Wesley, 1993.

[15] IEC 1508, *Functional Safety: Safety-Related Systems*, Draft, 1995.

[16] IEC 56(Sec)372, *Guide to test methods for dependability assessment for software*, 1994.

[17] IEC 880, *Software for Computers in the safety systems of nuclear power stations*, 1986.

[18] IEC/ISO 9126, *Information Technology - Software Product Evaluation - Quality Characteristics and guidelines for their use.*

[19] IEEE 1003.3, *Standard for Information Technology - Test methods for measuring conformance to POSIX.*

[20] ISO 12207, *Information Technology - Software life cycle processes*, 1995.

[21] ISO/IEC 12119, *Information Technology - Software Packages - Quality Requirements and Testing*, 1994.

[22] ISO 9000-3, *Quality Systems Guide to the Application of ISO 9001 to the Development, Supply and Maintenance of Software*, 1991.

[23] MIL-STD-498, *Software Development and Documentation*, 1992.

[24] Myers, G., *The Art of Software Testing*, Wiley Interscience, 1979.

[25] Nash, S.H. and Redwine, S.T., 'Map of the World of Software-Related Standards, Guidelines, and Recommended Practices', *Computer Standards and Interfaces*, Vol. 6 No. 2, pp. 245-265, 1987.

[26] prEN 50128, *Railway Applications - Software for Railway Control and Protection Systems*, Draft European Standard, 1995.

[27] RAC CRTA-TEST, *Testability Design and Assessment Tools.*

[28] RAC/CSA Q396, *Software Quality Assurance Program Standards.*

[29] Reid, S. C., 'The BCS Software Component Testing proto-Standard', *Proc. of the Second International Conference on Software Quality Management*, Edinburgh, July 1994.

[30] Reid, S. C., 'Popular Misconceptions in Module Testing', *Proceeding of the 13th International Conference on Testing Computer Software*, Washington DC, June 1996.

[31] Reid, S. C., 'An Empirical Analysis of Equivalence Partitioning, Boundary Value Analysis and Random Testing', *Proc. of the 4th International Software Metrics Symposium*, Albuquerque, November 1997.

[32] Reid, S. C., 'Module Testing Techniques - which are the most effective? Results of a Retrospective Analysis', *Proc. of the 5th European International Conference on Software Testing, Analysis and Review*, Edinburgh, November 1997.

[33] RTCA DO-178B, *Software Considerations in Airborne Systems and Equipment Certification,* RTCA, Dec 1, 1992.

[34] Tripp, L. and Voldner, P., 'A Market-Driven Architecture for Software Engineering Standards', *Proc. of the Software Engineering Standards Symposium,* Montreal, Quebec, August 1995. IEEE Computer Society Press, 1995.

[35] Wichmann, B., 'Problems and Strategies for Software Component Testing Standards', *Journal of Software Testing, Verification and Reliability*, Vol. 2, pp. 167-185. 1992.

[36] Wichmann, B., 'Why are there no measurement standards for software testing?', *Computer Standards and Interfaces*, Vol. 15, pp 361-364, 1993.