Accessible, rigorous measurement and test methods are key to creating quality software and increasing IT market competition. The authors describe here how their work at NIST contributes to these goals.

Toward Credible IT Testing and Certification

Shukri A. Wakid, D. Richard Kuhn, and Dolores R. Wallace
National Institute of Standards and Technology

To many, the prospect of software certification seems dubious. Software vendors promise that disks are free of defects, but state emphatically that they can't guarantee that the software on them is defect-free, nor that it is suitable for any purpose whatsoever. When it comes to determining whether a software product is dependable, safe, and effective, consumers are largely on their own.

The open availability of credible measurement and test methods is an important step toward assuring the quality of software-based systems and promoting competitiveness in the information technology (IT) market. Many national and international organizations are now working toward this goal, including national metrology institutes in the European Community and Japan, and industry groups such as Open Group (X/Open) and Underwriters Laboratories.

At the US National Institute of Standards and Technology's Information Technology Laboratory, our work focuses on establishing comprehensive certification capability for the IT industry. The NIST ITL approach uses the principles of measurement science, adapting them to measuring software product's conformance to particular standards, as well as its performance and dependability. ITL also works with in-

0740-7459/99/$10.00 © 1999 IEEE

Table 1
examples of basic measurement principles for software.

References	Measurement method	Uncertainty	NIST Examples
Specification or standard	Test cases t defined such that $P(t) \Rightarrow S(t)$, for implementation P and specification S.	Behavior conforming to the reference and options the reference permits. No guarantee of correctness, but some relationship to the reference correctness (depending on test-case diversity).	VRML, ATM, ISDN conformance tests; software aspects of Security Requirements for Cryptographic Modules (FIPS 140-1).
Reference implementation (code on a given platform)	Interoperation between implementation under test and reference implementation.	Conformance is limited by test-case diversity. Dependability may depend on reference correctness.	IPv6 protocols, IPSEC, RBAC, OCR code, ZPRIZE.
Standard reference data	Application of test software to data.	Output should correspond to "known" answers. Uncertainty is based on the comprehensive design and diversity of reference data. Performance and assurance can be evaluated.	Speech, text, and image corpus and data sets for mathematical and statistical software.
Known faults	Test cases t to detect the existence of known faults inserted in specification S.	Assurance against faults is limited by coverage of known faults.	Fault-based testing using model checking. Handbook and repository of software errors.
Criteria for assurance levels	Use various methods to interpret and test each criterion.	Assurance statements are as good as the criteria's completeness and proper interpretation. Behavior and functions not captured by the criteria can jeopardize assurance statements.	Common Criteria, Security Requirements for Cryptographic Modules (FIPS 140-1).

dustry to establish credible, cost-effective test suites to demonstrate software conformance to particular standards. ITL then issues these suites to accredited test laboratories, certified by either the NIST-administered National Voluntary Laboratory Accreditation Program (NVLAP) or by the private sector.

Here, we describe NIST's work, focusing on principles of measurement science and how they can be adapted for software. We also describe the use of such principles in international accreditation of software testing laboratories and their certification programs. The methods we discuss rely on standards, reference materials, or experience. But technology advances and changes almost daily. Nonetheless, these approaches can be applied to evolving technology so that the standards, reference materials and data are available as the technology matures. These methods may not yet provide 100 percent certification, but we believe they are a necessary route to that goal.

Measurement Principles

Software measurement science should use the same basic principles as physical measurement science, which requires a *reference*, a *measurement method*, and an *uncertainty statement*. At NIST, we identified different types of references, measurement methods, and uncertainties depending on the type of software being tested and the attributes being measured. Table 1 shows examples from each type.

Reference tracing

Traceability relates a measurement to an appropriate national standard "through an unbroken chain of comparisons."[1] In the US, government regulations and commercial contracts often require contractors to verify traceability in their measurements and support this by providing proof that their measurement equipment has been calibrated by laboratories or testing facilities that form part of this "unbroken chain."

In essence, traceability ensures that measurements are a reasonably accurate representation of the measured quantity. For software, traceability might seem unproblematic as software tests can be copied with 100 percent accuracy and do not require measurement-equipment calibration. But traceability is not as simple as it may appear.

To measure standard conformance, the reference is the standard itself. However, two different test sets measuring software standard conformance can produce different answers because of imprecision in the standard or the size of the sampling space. Typically, software standards are presented in natural language, leaving room for differing interpretations. In addition, software's discontinuous nature produces a measurement sampling space that is usually too large to completely evaluate. To reduce these prob-

Table 2
Reference implementations

Performance Testing	
Text retrieval test collections	http://www.nist.gov/itl/div894/894.02/products.html
TREC test collections on CD-ROM	http://trec.nist.gov/data/docs_eng.html
Speech processing evaluations and benchmark tests	http://www.nist.gov/speech/online.htm
Benchmark tests	http://www.nist.gov/speech/test.htm
Optical character recognition (OCR)	http://www.nist.gov/itl/div894/894.03/ocr/ocr.html
OCR test material on CD-ROM	http://www.nist.gov/itl/div894/894.03/databases/defs/vip_dbases.html#ocrlist
Fingerprint classification and matching	http://www.nist.gov/itl/div894/894.03/fing/fing.html
Fingerprint test data on CD-ROM	http://www.nist.gov/itl/div894/894.03/databases/defs/vip_dbases.html#finglist
Face recognition	http://www.nist.gov/itl/div894/894.03/face/face.html
Mug shot and face test data on CD-ROM	http://www.nist.gov/itl/div894/894.03/databases/defs/vip_dbases.html#facelist
SciMark (a benchmark for numeric-intensive applications in Java)	http://math.nist.gov/scimark/
S-Check tools	http://cmr.ncsl.nist.gov/scheck/scheck.html
MultiKron instrumentation boards and toolkits	http://cmr.ncsl.nist.gov/multikron
Dependability Testing	
Cryptographic modules and algorithms (specifications, tests, and validated implementations)	http://csrc.nist.gov/cryptval
Guide to available mathematical software	http://gams.nist.gov/
The Matrix Market (test data for comparative studies of numerical linear algebra algorithms)	http://math.nist.gov/MatrixMarket/
Statistical reference data sets	http://www.itl.nist.gov/div898/strd
MicroMagnetic modeling (standard problems to compare micromagnetic modeling codes)	http://www.ctcms.nist.gov/~rdm/mumag.org.html
Common Criteria	http://niap.nist.gov http://csrc.nist.gov/cc
Error fault and failure data	http://hissa.nist.gov/effProject/

lems, NIST plans to adapt software-engineering technology to conformance testing, addressing the precision problem using formal specifications and the sampling problem using statistical methods.

Measurement methods

Measurement methods vary with the scientific or technological field. Physical scientists typically build measurement methods on basic unit measurements that they can accurately measure, such as time and length. For example, a meter is rather precisely defined as "the length of the path traveled by light in a vacuum during a time interval of 1/299 792 458 of a second."[1]

Engineers typically measure software by executing it on a given data set, but they can also use other methods. For example, function points can measure development effort, and the cyclomatic number or similar metrics are sometimes used to estimate software maintenance or testing complexity.

Uncertainty statements

Historically, researchers have used many different approaches to evaluate and express the uncertainty of physical measurement results. In 1977, the International Bureau of Weights and Measures, in collaboration with other metrology institutes, proposed a specific solution to this inconsistency. This proposal produced a recommendation to describe a result's uncertainty using two categories.[2] The first, Type A, are those measurements evaluated by statistical methods; type B are those evaluated by other means. Thus, statistical variances can be estimated directly for category A; those in category B can be characterized by approximations to the assumed corresponding variances. The uncertainties can then be combined in various ways, depending on the quantity being measured.

Uncertainty is a challenge for software measurement, both in how to reduce it and how to describe the uncertainty that inevitably remains. With parallel and distributed applications, measuring ap-

plication performance is particularly difficult. Collecting performance measurement data adds 10 to 400 percent to a program's execution time, which can change execution characteristics on parallel and distributed programs.

To address this, NIST developed the MultiKron VLSI instrumentation chips and interface boards, which capture performance data of high-speed parallel processors and workstations by recording events triggered either by software memory writes or hardware signal transitions. The chips can either timestamp captured data and send it over a collection network or use it to control chip counters and clocks. The resulting measurements are accurate and give researchers insight into the source of performance bottlenecks. They can therefore learn how to scale system designs upwards without significantly perturbing the system. NIST also developed the S-Check tool, which perturbs parallel software to determine performance bottlenecks. Existing statistical methods for estimating reliability require knowledge of the input distribution, another source of uncertainty in measurement. Thus, a particular company's method for estimating reliability for an individual product may be unsuitable for other similar products, because input distribution can vary widely.

Reference implementation

NIST has worked with industry and academic researchers to develop reference implementations that are defined by standards, tested by certifiable test methods, and traceable to standards. These implementations are available to organizations to assess their own measurement methods or assign test-method values.[3] Example implementations include those developed for IP version 4, role-based access control, and Z39.50 search protocols. NIST developed some tests, often with industry cooperation, while other tests were developed by industry based on national laboratories' test criteria. Table 2 shows URLs for the performance and dependability tests.

Providing common tests is only part of the answer to software certification. An efficient, market-driven testing infrastructure for information technology also requires internationally acceptable procedures for both accrediting test laboratories and mutually recognizing their results.

The NVLAP Program

Public testing technology gives vendors criteria so they can self-certify their products as compliant with a known measurement technology. Third-party commercial testing laboratories can also use this public technology to meet user-group requirements. Certification is not a guarantee against failure, only a statement about risk. For the certification process to work, there must be credible and cost-effective tests available, clearly defined testing methods, and standardized reporting formats. Further, user organizations must promote and require product certification.

To this end, NIST administers the National Voluntary Laboratory Accreditation Program, or NVLAP, a series of laboratory accreditation programs. Each LAP includes specific calibration and test standards, as well as methods and protocols to satisfy accreditation needs in a particular area.

Accreditation process

When a laboratory applies for accreditation, NVLAP evaluates its technical qualifications and competence to carry out specific calibrations or tests. In information technology, for example, NVLAP accredits laboratories testing against FIPS 140-1 (cryptographic modules), GOSIP OSI profiles, MIL-STD-462 (Tempest), and the IEEE Posix operating-system interface. As Tables 3 and 4 show, NIST's ITL transfers measurement technology to private commercial organizations and assists in the development of private testing services.

An efficient, market-driven testing infrastructure requires internationally acceptable procedures for both accrediting test laboratories and recognizing their results.

As part of NVLAP's Procedures and General Requirements, the US Code of Federal Regulations (CRF, Title 15, Part 286) publishes accreditation criteria that encompass ISO/IEC Guide 25 and ISO 9002 requirements. NVLAP grants accreditation after an organization successfully completes a process that includes application and fee payment, on-site assessment, deficiency resolution, proficiency testing, and technical evaluation. The user organization oversees the certification process. NVLAP accreditation is available to US public and private laboratories; laboratories outside the US can be accredited if they

Table 3
Operational software testing services

Test Service	Start Date	Termination Date	Certificate Offered by	Testing Offered by
SQL	1990	1 July 1997	NSTL, Terwilliger, EDS	NSTL, Terwilliger, EDS
POSIX	1991	31 Dec. 1997	IEEE, X/Open	Mindcraft, Perennial, X/Open
Ada	1985	July 1997	AJPO	AJPO Recognized Test Labs
Fortran 78	1979 (GSA), 1986 (NIST)	7 June 1998	EDS	EDS
Cobol 85	1974 (US Navy), 1986 (NIST)	7 June 1998	EDS	EDS
C	1989	on or before 1 Oct. 1998	EDS and/or Perennial	EDS and/or Perennial
CGM	1994	1 Oct. 1998	ATA	ATA Recognized Test Labs

Table 4
New testing services developed with NIST assistance

Test Service	Start Date	Sponsored by	Type of Service	Certification
VRML	1997	VRML Consortium	Browser testing using NIST VTS. VRML content testing using NIST Viper	VRML Consortium is discussing a certification program
Spatial Data Transfer	1998	US Geological Survey	Test implementation (encoders, decoders, transfers) of SDTS	Plans for EDS to issue certificates and do testing
IMS	1988	Educause	Test implementations of IMS specifications and prototypes	Working with IMS to develop a framework for a certification program

meet the standard requirements and pay travel fees.

NVLAP evaluates and recognizes performance and offers laboratories expert technical guidance to upgrade performance. Accreditation signifies that a laboratory meets NVLAP requirements in the following areas: accommodation and environment; calibration and test methods; certificates and reports; complaints; equipment and reference materials; measurement traceability and calibration; organization and management; outside support services and supplies; personnel; quality system, audit, and review; records; and subcontracting. NVLAP accreditation does not guarantee laboratory performance or test/calibration data; it is solely a finding of laboratory competence. A laboratory can cite its accredited status and use the NVLAP logo on reports, stationary, and in business and trade publications provided that its use does not imply product certification.

Vendor and consumer benefits

NIST accredits laboratories to provide testing services under the NVLAP program to increase competition for accreditation services, and thereby increase testing availability and reducing its costs. Metrology institutes in other countries provide similar accreditation services. Some government and industry groups also offer test-result validation. They do this by reviewing accreditation test results to ensure tests were run properly. Tested products are then added to a validated products list that is available to consumers.

Third-party laboratories must meet specific criteria to be accredited to conduct tests. Industry consortia or national and international standards bodies can establish these criteria.

Vendors have one of three options when seeking accreditation.

- *Self-declaration.* In some cases, vendors declare their own compliance with specified requirements. Consumers who take the vendor's claims at face value can avoid the cost of third party testing.
- *Third-party evaluation.* Vendors can submit products for evaluation by third-party laboratories, which in turn provide testing results to consumers. Consumers judge test validity.
- *Third-party evaluation with government or industry validation.* For some products, consumers

Table 5
Conformance tests and reference data

Fortran78	http://www.itl.nist.gov/div897/ctg/fortran_form.htm
Cobol85	http://www.itl.nist.gov/div897/ctg/cobol_form.htm
CGM	http://www.itl.nist.gov/div897/ctg/cgm_form.htm http://www.itl.nist.gov/div897/ctg/graphics/cgmv3hd.htm
PHIGS	http://www.itl.nist.gov/div897/ctg/phigs_form.htm
RDA	http://www.itl.nist.gov/div897/ctg/rda_form.htm
Java conformity assessment and diagnostics	http://www.nist.gov/java_ca.htm
VRML conformance tests and viper reference parser	http://www.nist.gov/vrml.html
SQL	http://www.itl.nist.gov/div897/ctg/sql_form.htm
Posix	http://www.itl.nist.gov/div897/ctg/posix_form.htm
Role-based access control	http://hissa.ncsl.nist.gov/rbac/
NIST integrated services protocol instrument	http://www.antd.nist.gov/antd/html/ispi.html
Cryptographic modules and algorithms (specifications, tests, and validated implementations)	http://csrc.nist.gov/cryptval

may want evidence of an accurate, complete evaluation. In this case, vendors can have their test results validated by a government or industry organization, which offers consumers added security.

Accredited-laboratory certification can give vendors a significant advantage with customers and can help boost exports. In April 1999, five countries—the US, Canada, Germany, the UK, and France—signed the Common Criteria Mutual Recognition Agreement for testing security products; many other nations in Europe and the Pacific rim are expected to follow. Under the agreement, signatory nations will recognize security evaluations from each other's accredited testing laboratories. Vendors who have their products tested by certified labs can also avoid the cost of repetitive testing, which reduces time to market and makes it easier for signatory countries to sell their products. At present, accredited laboratories provide testing for conformance to various international standards and dependability tests in the security field.

Testing Areas

Software conformance, assurance, and performance are three key attributes of its quality. For each of these attributes, the metrology—reference, measurement method, and uncertainty statement—varies considerably.

Conformance testing

In conformance testing, the reference is the standard or specification; the measurement method prescribes a test configuration, a platform type, and test cases. For example, we might test conformance to the IEEE Posix 1003.1 OS kernel-interface standard, configured with specified options and running on a particular platform. In this case, the evaluation typically yields a binary result: either the software conforms to the options tested or it doesn't. A statement of uncertainty is also necessary, indicating that the conformance statement refers only to this particular combination of configuration options, platform, and test suite. Although conceptually simple, the uncertainty statement can be complicated by the numerous options and alternatives typically offered with software products.

To test products against standards, open, well-understood tests are crucial. Conformity testing provides a systematic examination of how a product, process, or service fulfills specified requirements.[4] Such testing does not rule on diagnostics or dependability, and may not even rule on conformity if the specification is not rigorously defined. Also, many of today's IT systems are built from one or more commercial, off-the-shelf components, each of which requires conformance testing.

NIST works with industry organizations to develop conformance tests for vendor products that have standards or rigorous specifications, such as compilers. NIST has developed test suites for older languages such as Fortran, Cobol, and SQL. More recently, ITL has been actively developing tests for modern technologies such as Java, VRML, and firewalls. ITL has also developed conformance tests for security clauses in the FIPS PUB 140-1, Security Requirements for Cryptographic Modules.[5] Table 5 shows URLs for conformance tests and reference data.

Conformance testing against a specification that may or may not be a standard is a common form of testing. Such testing does not decisively conclude diagnostics or assurance, and may not provide a binary statement about conformance, particularly if the specification is vaguely defined. Many developers use these standard test suites during develop-

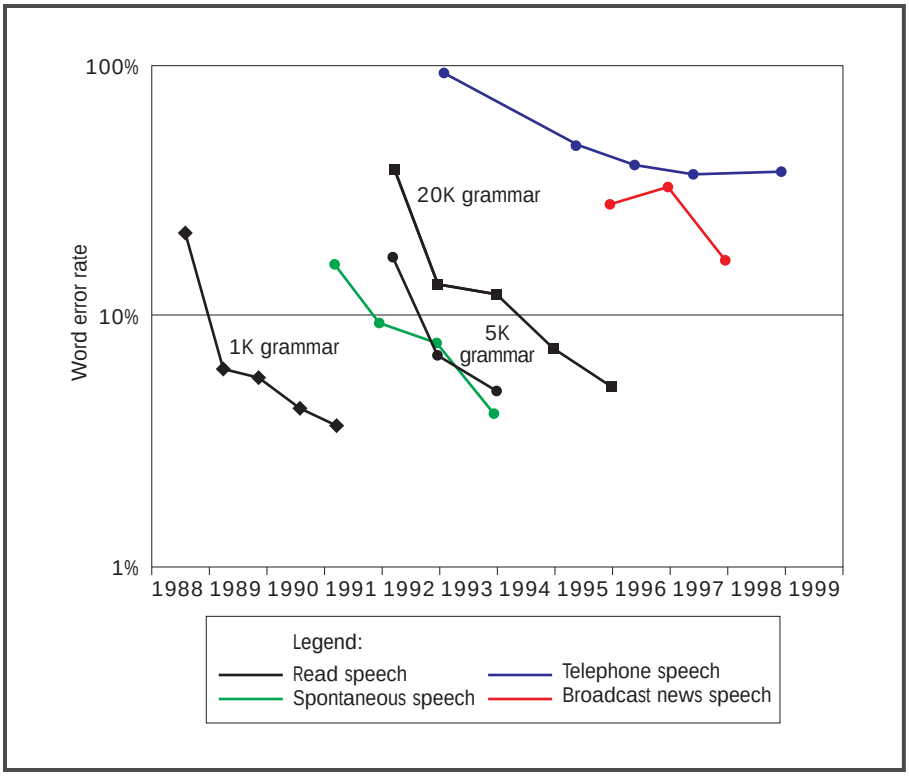


Figure 1. Error-rate improvement for speech recognition systems using the NIST speech-recognition corpus. Like most standard reference material, the speech-recognition corpus often comes with an evaluation method and scoring protocol to help researchers determine uncertainty.

ment to get continuous feedback on product quality. For example, some VRML web browsers have been developed using periodic testing against NIST's specification-based VRML test suite. NIST has developed other specification-based tests for SQL, Posix, compiler testing, asynchronous transmission mode, and cryptomodule testing (FIPS 140-1).

Performance testing

For software performance, the reference is often a representation of a typical software user's data. A reference for a transaction processing system, for example, might be a benchmark data set representing a given transaction mix for a group of concurrent users, with performance stated in a form such as "60,000 transactions per minute with 55,000 concurrent users on the *tpmXYZ* benchmark."

Other types of performance testing use a standard reference material. For example, speech, text, and image collections have been used to evaluate algorithmic performance for speech understanding, text retrieval, and image recognition. Such references are given to researchers with an evaluation method and a scoring protocol that determines uncertainty. For example, Figure 1 shows error-rate improvements for speech recognition systems that process different speech types within the NIST speech-recognition corpus. This corpus is widely used by industry, particularly within the DARPA research community.

Dependability testing

One way to improve dependability is to evaluate a system's formal specification against a set of formally defined requirements, producing a formal or semi-formal proof. Source code and tests are derived from the formal specification, providing traceability to the requirements. This type of assurance tends to be expensive, but is often required for the highest evaluation levels in national and international standards, such as the Security Requirements for Cryptographic Modules (FIPS 140-1) and the international Common Criteria for security products.

A second method examines or tests the software product against reference data for a class of products and, if possible, for a specific class architecture. Measurement methods for this approach include analytic techniques, such as inspection, static analyses with automated tool support, and all types of testing designed to find problems within a specific fault class. Computing the uncertainty in fault-finding testing requires numerous statistical techniques.

ITL is collecting reference data on software faults and failures to help with fault-finding activities.[6] The project has produced:

- a web tool to assist the industry in collecting and analyzing fault and failure data for individual projects;
- a publicly accessible repository of project data that users can sort according to attributes, with links to statistics and graphics;
- frequency profiles of fault classes for application domains and specific architectures within them; and
- a handbook of fault types associated with prevention and detection methods.

Each handbook chapter classifies faults discovered during development or maintenance according to application domain, architecture, language, and special interests such as safety or security. The handbook will also address system failures discovered during operation. Industry, federal agencies,

Table 6
Automated Test-Generation costs

	Traditional	Formal spec and verification without test generation	Formal spec and verification with test generation	Formal spec with test generation	Formal spec with test generation
Design, code, and other costs	50	50	50	50	50
Test coding	30	30	15	15	10
Test execution	20	20	20	20	20
Formal specification	—-	10	10	10	10
Formal verification	—-	10	10	—-	—-
Comparative costs	100%	120%	105%	95%	90%

and universities are providing ITL with handbook data under nondisclosure agreements. ITL is continually seeking additional contributors.

Such reference data is valuable to both developers and accredited laboratories. Developers can use the information to build quality processes around known fault types. They can also use frequency profiles and statistical methods to assess the uncertainty in fault removal. Accredited laboratories can use the data to develop test sets for products against specific fault classes.

Finally, diagnostic testing, which looks for bugs in implementations, can be considered a form of dependability testing. NIST has a diagnostic tool named Vmview that identifies bugs in Java implementations.

Research Issues

Ensuring software dependability is difficult, not only because of its size and complexity, but also because the source code for COTS and other externally developed components is often unavailable. Improved methods of assurance are thus essential for complex component-based systems. Defining reference standards precisely is crucial to providing rigorous assurance; formal specifications provide the best precision for standards, and formal methods will be increasingly important as the IT industry moves toward greater use of standardized, off-the-shelf components.

The formal-specification development process itself is often as effective at finding errors as the verification effort in which that spec is used. Developing formal specifications requires a detailed and precise system understanding, which helps expose errors, ambiguities, and omissions. Yet despite their advantages, formal specifications are rarely used, as they require highly skilled developers and are viewed as not cost effective.

Formal methods were developed to rigorously analyze system properties, a task that requires precise system descriptions. In practice, developers sometimes use formal specifications to show system conformance to formal requirements. The specification can also be used to implement the system in code. Thus, the cost of specification development and proof must be less than the cost of allowing faults to remain in released products.

But developing rigorous system tests also requires a precise, complete description of system functions, and practical system assurance requires testing, even when formal methods are used. Test development is typically an enormous expense, often up to half of total development cost. Thus any increases in the efficiency of test development can have a significant impact on product cost.

Although not widely used in system assurance, formal verification adds costs beyond system testing. Formal verification costs have two components: formal system specification development and analysis of the specifications in relation to requirements. In the latter case, the analysis might be a computer-assisted proof or an automated verification through model checking. Although this type of formal verification may reduce the total system cost, it can add 10 to 20 percent or more on upfront costs.

However, formal specifications have value beyond analysis and proof. They can generate complete test cases, with input data and expected results. This results in a dramatic reduction in testing costs. Table 6 shows estimated system development costs using different specification configurations.

To date, most research on automated software testing has focused on structural testing, which is testing based on execution paths for code with a specified function. However, if source code is unavailable, structural testing is impossible. An alternative is to use specification-based testing, in which tests are derived from the specification alone. A new ITL project is developing a test-generation tool that can automatically generate complete test cases from formal specifications.[7] In this project, faults are in-

serted into a specification, and a model checker generates counterexamples that can be post-processed into complete test cases in Java. Initial results show test coverage to be as good or better than handcrafted tests.

Generating tests from specifications can make formal methods cost effective for a much larger class of systems. In the US, formal methods use is largely confined to secure or safety-critical systems—those systems whose failure can have catastrophic cost. But if the costs of formal methods can offset the possibly higher cost of test development, formal techniques become much more attractive. ❖

References

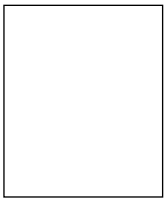
1. *ANSI/NCSL Z540-1-1994*, American Nat'l Standards Inst., New York, 1994.
2. *Recommendation INC-1*, Working Group on the Statement of Uncertainties, Bureau International des Poids et Mesures, Sévres, France, 1980.
3. *Metrology for Information Technology*, National Institute of Standards and Technology, New York, 1997.
4. *ISO/IEC Guide 2:1996, Standardization and Related Activities—General Vocabulary,* Int'l Standards Organization, Geneva, 1996.
5. *Security Requirements for Crytographic Modules*, FIPS PUB 140-1, US Dept. Commerce, Nat'l Inst. of Standards and Technology, New York, 1994.
6. D.R.Wallace, "Enhancing Competitiveness Via a Public Fault and Failure Data Repository," *Proc. IEEE High-Assurance Systems Eng. Symp.*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1998.
7. P.E. Ammann, P.E. Black, and W. Majurski, "Using Model Checking to Generate Tests from Specifications," *Proc. IEEE Int'l Conf. Formal Engineering Methods*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1998.

Readers may contact Kuhn at NIST, Bldg. 820, Room 562, Gaithersburg, Md. 20899; e-mail kuhn@nist.gov.

About the Authors

Shukri A. Wakid is the chief information officer at the National Institute of Standards and Technology, and past director of NIST's Information Technology Laboratory, where he managed many programs, including those in advanced network technologies, computer security, information systems architecture, statistics, high performance systems and services, and information services. Shukri is a senior member of the IEEE, vice chair of the Technical Advisory Board of the IEEE Computer Society, and a member of the advisory board of the Instructional Management System of the Educause consortium.

Richard Kuhn is manager of the Software Quality Group at NIST. He has published more than 30 papers on software testing, analysis, and assurance, and on security and standards for open distributed systems. From 1994 to 1995, he was program manager for President Clinton's Information Infrastructure Task Force's Committee on Applications and Technology. Before joining NIST in 1984, he worked as a systems analyst with NCR Corporation and the Johns Hopkins University Applied Physics Laboratory. Kuhn received an MS in computer science from the University of Maryland at College Park, and an MBA from the College of William and Mary.



Dolores R. Wallace leads ITL's Reference Data: Software Fault and Failure Data Collection Project, which provides metrology and reference data for software assurance. Her research interests include software technology measurement and experimentation. She is coauthor of Software Quality Control, Error Analysis, and Testing (Noyes Data Corporation, 1995) and cochair of the IEEE STD 1012 -1998, Software Verification and Validation. Wallace received an MS in mathematics from Case Western University.