IEEE Test and Diagnostics Standards

John Sheppard
ARINC
2551 Riva Road
Annapolis, MD 21401
410-266-2099
jsheppar@arinc.com

Mark Kaufman
NWAS
PO Box 5000
Corona, CA 91718
909-273-5725
kaufmanma@corona.navy.mil

Abstract - The 1232 family of standards were developed to provide standard exchange formats and software services for reasoning systems used in system test and diagnosis. The exchange formats and services are based on a model of information required to support test and diagnosis. The standards were developed by the Diagnostic and Maintenance Control (D&MC) subcommittee of IEEE SCC20. The current efforts by the D&MC are a combined standard made up of the 1232 family, and a standard on Testability and Diagnosability Metrics, P1522. The 1232 standards describe a neutral exchange format so one diagnostic reasoner can exchange model information with another diagnostic reasoner. In addition, software interfaces are defined whereby diagnostic tools can be developed to process the diagnostic information in a consistent and reliable way. The objective of the Testability and Diagnosability Metrics standard is to provide notionally correct and mathematically precise definitions of testability measures that may be used to either measure the testability characteristics of a system, or predict the testability of a system. The end purpose is to provide an unambiguous source for definitions of common and uncommon testability and diagnosability terms such that each individual encountering it can know precisely what that term means. This paper describes the 1232 and P1522 standards and details the recent changes in the information models, restructured higher order services and simplified conformance requirements.

INTRODUCTION

Tools such as Computer Aided Design (CAD), Computer Aided Manufacturing (CAM), and Automatic Test Equipment (ATE) are generating large collections of product data. One characteristic of product data is object-like structure. Product databases are large due to the complexity of the products themselves and the detail of the information. Product databases are often difficult to maintain because "corporate knowledge" often evaporates as the product transitions from design to manufacturing, from manufacturing to the market, and finally to repair. Product description at one phase of the lifecycle must be useable by tools at subsequent phases to avoid re-entering data. Information is expensive to obtain. Each phase of the product cycle has unique information needs and makes unique information contributions. Better decision-making requires better information in a timely manner. Data mining warehousing, XML, information systems, all add value. This trend has been accelerating and now touches our everyday lives. Do you have a grocery club card?

The Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE) standards are product information exchange standards for test and diagnosis. The original standards, the 1232 series, developed a means of exchange of information between diagnostic reasoners. As the information models for the 1232 standards were developed, it became apparent that these models could be used for standardizing testability and diagnosability metrics.

IEEE Std 1232-1995 defines the architecture of an AI-ESTATE-conformant system and has been published as a "full-use" standard;. However, this standard was published before the vision of AI-ESTATE was fully developed. IEEE Std 1232.1-1997 defines a knowledge and data exchange standard and was published as a "trial-use" standard. Trial-use indicates that it is preliminary in nature, and the standards committee is seeking comments from organizations attempting to implement or use the standard. In 1998, IEEE Std 1232.2-1998 was approved. This standard formally defines a set of standard software services to be provided by a diagnostic reasoner in an open-architecture test environment. Since IEEE Std 1232.2-1998 is also a trial-use standard, comment and feedback are solicited. These standards were developed using information modeling. Five information models addressing static and dynamic aspects of the diagnostic domain were developed. The IEEE 1232 AI-ESTATE series of standards also provide the foundation for precise and unambiguous testability and diagnosability metrics.

As systems became more complex, costly, and difficult to diagnose and repair, initiatives were started to address these problems. The objective of one of these initiatives, testability, was to make systems easier to test. Early on, this focused on having enough test points in the right places. As systems evolved, it was recognized that the system design had to

0-7803-6395-7/00/$10.00 ©2000 IEEE

include other characteristics to make the system easier to test. As defined in MIL-STD-2165, testability is "a *design characteristic* which allows the status (operable, inoperable, or degraded) of an item to be determined and the isolation of faults within the item to be performed in a timely manner." [1]. The purpose of MIL-STD-2165 was to provide uniform procedures and methods to control planning, implementation, and verification of testability during the system acquisition process by the Department of Defense (DoD). It was to be applied during all phases of system development—from concept to production to fielding. MIL-STD-2165, though deficient in some areas, provided useful guidance to government suppliers. Further, lacking any equivalent industry standard, many commercial system developers have used it to guide their activities although it was not imposed as a requirement.

A VISION FOR TEST AND DIAGNOSIS STANDARDS

Diagnosis

The vision of AI-ESTATE standards is to provide an integrated, formal view of diagnostic information as it exists in diagnostic knowledge bases and as it is used (or generated) in diagnostic systems. We assert that the whole purpose of testing is to perform diagnosis [2]. In justifying this assumption, we rely on a very general definition of diagnosis, derived from its Greek components (δια γιγνωσκειν) meaning, "to discern apart." Given such a broad definition, all testing is done to provide information about the object being tested and to differentiate some state of that object from a set of possible states.

In support of this vision, the Diagnostic and Maintenance control (D&MC) committee has been working on combining the existing 1232 standards into a single, cohesive standard. This "unified" standard provides formal specifications of all of the information models (both for file exchange and for diagnostic processing), from which the service specifications are then derived. The architectural framework is retained at the conceptual level to emphasize that a wide variety of implementation models are possible that still support standard exchange of information as long as the definition of that information is clear and unambiguous. Thus, in a sense, the models define the architecture, and the implementation is left entirely to the implementer.

With this vision in mind, we expect AI-ESTATE to play a central role in any test environment (thus the "All Test Environments" part of the name). To date, the focus of the standards has been the development of specifications supporting diagnosis in the traditional sense of the word (i.e., fault isolation). The broader AI-ESTATE vision involves tying diagnostic information to explicit product behavior descriptions, assessments of the ability of testing to satisfy its requirements, and maturation of the diagnostic process through test and maintenance information feedback.

Testability and Diagnosability

In 1997, the AI-ESTATE committee began to work on a new standard to take over where the canceled MIL-STD 2165 left off. The military standard focused on specifying the essential elements of a testability program and explained the elements needed to define a testability program plan. In addition, MIL-STD 2165 contained the "definition" of several testability metrics, and included a testability checklist to be used to determine overall system testability. The approach being taken to develop this new standard involves defining testability and diagnosability metrics based on standard information models. The updated models in the combined 1232 standard provide a solid basis for the development of testability and diagnosability metrics.

THE AI-ESTATE ARCHITECTURE

According to IEEE Std 1232-1995, the AI-ESTATE architecture is "a conceptual model" in which "AI-ESTATE applications may use any combination of components and intercomponent communication" [3]. Another "view" of the AI-ESTATE architecture is provided by IEEE Std 1232.2-1998. IEEE Std 1232.2-1998includes explicit definitions of information services to be provided by a diagnostic reasoner, where the services "can be thought of as responses to client requests from the other components of the system architecture" [4]. More specifically, "each of the elements that interface with the reasoner will interact through [an] application executive and will provide its own set of encapsulated services to its respective clients" [4]. Figure 1 illustrates this architecture.

Although not necessarily obvious from the standards themselves, these two "views" of the AI-ESTATE architecture present an interesting dichotomy. Specifically, the architecture standard provides a concept of AI-ESTATE that permits any communication mechanism to be used between components of a test environment in support of the diagnostics provided by that environment. The service specification, on the other hand, seems to cast the communication mechanism in the form of a client-server architecture. We note that the service specification did not intend to require a client-server approach but presented this as an example architecture that fits within the component orientation.

We note that the intent of AI-ESTATE is to provide a formal, standard framework for the exchange of diagnostic information (both static and dynamic) in a diagnostic environment. This exchange occurs at two levels. At the first

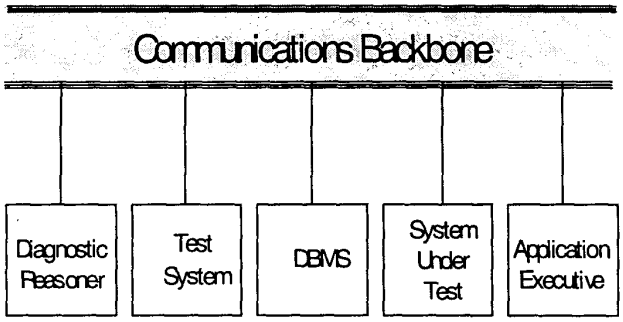

Figure 1. AI-ESTATE Architecture

level, model data and knowledge are exchanged through a neutral exchange format, as specified by IEEE Std 1232.1-1997 [5]. At the second level, specified by IEEE Std 1232.2-1998 [4] information is exchanged as needed between software applications within the diagnostic environment. This information includes entities from a model or information on the current state of the diagnostic process.

To facilitate encapsulation of the information and the underlying mechanisms providing that encapsulation, AI-ESTATE assumes the presence of an "application executive." We emphasize that this application executive need not be a physically separate process but can be identified as a "view" of the process when it involves the communication activity. In the following sections, we will provide a more detailed discussion of the exchange and service elements of the architecture.

Data and Knowledge Exchange

ISO 10303-11 (EXPRESS) and ISO 10303-21 (STEP Physical File Format) are used to define information models and exchange formats for diagnostic knowledge [6, 7]. These two international standards are being maintained by the STEP (Standard for the Exchange of Product model data) community. The current approach to static information exchange within AI-ESTATE is to derive the exchange format from the formal information models as specified in the ISO standards.

The purpose of information modeling is to provide a formal specification of the *semantics* of information that is being used in an "information system." Information models identify the key entities of information to be used, their relationships to one another, and the "behavior" of these entities in terms of constraints on valid values [8]. The intent is to ensure that definitions of these entities are unambiguous.

IEEE 1232.1 was published as a "trial-use" standard to provide a period for people to study it, attempt to implement it, and provide feedback to the AI-ESTATE committee on the ability of the standard to satisfy the stated requirements. Since publication, comments have been received to indicate that ambiguity still exists in the information models. Because of the concern that the information models are still ambiguous, the models have undergone close examination and modification.

Diagnostic Services

The approach taken to defining services in AI-ESTATE has been based on the traversal (i.e., the following of the relationships defined between model entities to access specific pieces of information in the models) of the information models. The "simplest" services involve traversing the models defined in IEEE 1232.1 (i.e., the exchange models); however, these models provide little functionality in terms of actual diagnosis.

In IEEE 1232.2, a novel use of information modeling was applied in that a dynamic information model was specified to support dynamic services. This model, called the "dynamic context model" (DCM) relied on dynamically creating entities that populate the model during a diagnostic session. In fact, as suggested by "*dcm.session*" and "*dcm.step*" in the model shown in Figure 2, a diagnostic session is modeled as a sequence of steps instantiated from the set of possible values specified in the static model. Details of how the service specification is expected to be implemented can be found in [9, 10].

One of the concerns raised by a member of the AI-ESTATE committee was whether the standard specifies a set of services or simply an Application Programming Interface. The claim was that the service specification must include a behavior specification as well and that this can only be accomplished by defining a set of baseline behaviors, perhaps through some sort of test bed.

The committee observed that there are different opinions over the difference between a service specification and an API specification. Many, in fact, took issue with the claim that they were different. Further, it was determined that including test cases to specify standard behavior was not desirable due to the wide variety of diagnostic approaches using common diagnostic knowledge. Rather, it was believed that it was more important for the information itself to be standardized and the specific behavior to be left to the implementation.

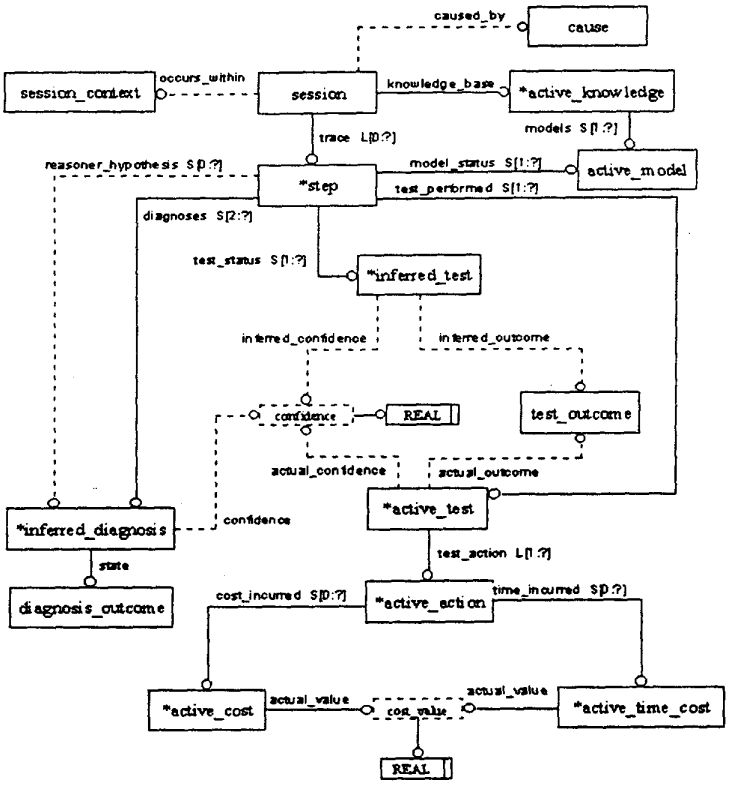

Figure 2 Dynamic Context Model

In the original definition of the services, the list was limited to the set of "primitive accessor" services corresponding to elements in the information models. Recently, the services have been redefined to provide a naming convention covering the primitives in the models (thus reducing the number of services to be defined from over 400 to approximately 160). In addition, two new classes of services have been added: utility services and higher-order services. The utility services provide methods for determining if "optional" model attributes exist and for counting/indexing elements within aggregate attributes (i.e., sets and lists). The higher-order services provide convenient methods for performing reasoning within the AI-ESTATE framework. These services include, for example, *apply_test_outcome* puts an outcome to the active test associated with the current step in the DCM. It then spawns the inference process within the attached reasoner. When the inference is complete, it proceeds to put inferred outcomes for related tests and diagnoses within the current step. As an option, it may also update the current hypothesis (if the reasoner is capable of generating a hypothesis).

Conformance

The published standards provide very strict rules for conformance. Specifically, data exchange must conform to the published models. No subsets are permitted (except by dropping optional attributes), and extensions must be handled via the EXTEND schema. In addition, all services must be specified (i.e., no extensions and no subsets).

Currently, the D&MC is debating about a more manageable and flexible approach to conformance. The current proposal requires an application to include a conformance "matrix" with associated documentation to identify those areas claiming to be conformant.

To claim minimal conformance to IEEE Std 1232 for model development, a conformance matrix containing at least the following must be provided. The core model elements of the common element model (CEM) plus at least one of either the Fault Tree Model (FTM), Diagnostic Inference Model (DIM) or Enhanced Diagnostic Model (EDIM). Further for each Enhanced Model Element the capability of the application to either read or write the element must be specified.

To claim minimal conformance to IEEE Std 1232 for the application runtime environment (i.e., services), a conformance matrix containing at least the following must be provided. The Core Primitive Services for the CEM and DCM and at least one of the Fault Tree Model (FTM), Diagnostic Inference Model (DIM) or Enhanced Diagnostic Model (DIM). The Enhanced Primitive Services for the CEM and DCM must be specified. The Core Higher-Order Services for the DCM are required. The Enhanced Higher-Order Services must be specified.

TESTABILITY AND DIAGNOSABILITY METRICS

Testability has been broadly recognized as the "-ility" that deals with those aspects of a system that allow the status (operable, inoperable, or degraded) or health state to be determined. Early work in the field primarily dealt with the design aspects such as controllability and observability. Almost from the start this was applied to the manufacturing of systems where test was seen as a device to improve production yields. The concept slowly expanded to include the aspects of field maintainability such as false alarms, isolation percentages, and other factors associated with the burden of maintaining a system.

In the industry, many terms such as test coverage and Fraction of Fault Detection (FFD) are not precisely defined or have multiple definitions. Further, each diagnostic tool calculates these terms differently and therefore the results are not directly comparable. Some measures, such as false alarm rate, are not measurable in field applications. Other measures such as Incremental Fault Resolution, Operational Isolation, and Fault Isolation Resolution appear different, but mean nearly the same thing.

Lacking well-defined testability measures, the tasks of establishing testability requirements, and predicting and evaluating the testability of the design are extremely difficult. This in turn makes effective participation in the design for testability process difficult. These difficulties will be greatly diminished by the establishment of standard testability metrics. An immediate benefit will come with a consistent, precise, and measurable set of testability attributes that can be compared across systems, tools, and within iterations of a system's design.

As we strive to establish concurrent engineering practices, the interchange between the testability function and other functions becomes even more important. To create integrated diagnostic environments, where the elements of automatic testing, manual testing, training, maintenance aids, and technical information work in concert with the testability element, we must maximize the reuse of data, information, knowledge, and software. Complete diagnostic systems include Built-In-Test (BIT), Automatic Test Equipment (ATE), and manual troubleshooting. It would be desirable to be able to predict and evaluate the testability of systems at these levels.

It is not an accident that the P1522 standard contains both the words testability and diagnosability. The distinction is not always easy to maintain, especially in light of the expansion of the use of the testability term. Diagnosability is the larger term and encompasses all aspects of detection, fault localization, and fault identification. Testability is a design characteristic of the system. The boundary is fuzzy and often it is not clear when one term applies and the other does not. The P1522 standard is meant to encompass both aspects of the test problem. Because of the long history of the use of the testability term, we will seldom draw a distinction. However, the use of both terms is significant in that testability is not independent of the diagnostic process. The writing of test procedures cannot and should not be done separately from testability analyses. To do so, would be meeting the letter of the requirements without considering the intent.

The objective of the P1522 standard is to provide notionally correct, inherently useful, and mathematically precise definitions of testability metrics and characteristics. It is expected that the metrics may be used to either measure or predict the testability of a system, . Notionally correct means that the measures are not in conflict with intuitive and historical representations. Beyond that, the measures must be either measurable or predictable. The former may be used in the specification and enforcement of acquisition clauses concerning factory and field-testing, and maintainability of complex systems. The latter may be used in an iterative fashion to improve the factory and field-testing and maintainability of complex systems. The most useful of all are measures that can be used for both. Because of the last point, the emphasis will be on measurable quantities (metrics).

Things that can be enumerated by observation and folded into the defined figures-of-merit will be developed into metrics. However, a few measures are inherently useful on the design side even if they are not measurable in the field and they are defined in a separate section in P1522. The end purpose is to provide an unambiguous source for definitions of common and uncommon testability and diagnosability terms such that each individual encountering the metric can know precisely what that metric measures.

Testability and Diagnosability Metrics and the 1232 Standards

Metrics are a measure of some identifiable quantity. The metrics of P1522 are derived from information obtained from the information models in IEEE 1232. The very basic information, the primitives, is counts of something. The number of faults, components, and functions are obtainable from the information services of the 1232 models. For example, a testability analysis tool would ask for the total number of faults and then ask for the number of faults detected. The tool would then calculate the fraction of faults detected. This example is extremely simplified.

In the revised CEM, each diagnosis has a criticality. This relationship allows the testability analysis tool to generate metrics that are based on Failure Effects Mode and Criticality Analysis (FEMCA). Adding this information to the previous example would generate the fraction of catastrophic faults detected. A further variation on this would be to determine the percentage of failed components that would lead to a catastrophic failure detected by Built-in-Test (BIT).

Metrics Issues

MIL-STD-2165 defined Fraction of Faults Detected (FFD) two ways. The first is the fraction of *all* faults detected by BIT/External Test Equipment (ETE). The second is the fraction of *all detectable* faults detected by BIT/ETE [1]. False alarms were excluded from the definition. From these two variations grew many others. As noted in "Organizational-Level Testability" [11] FFD has been

defined as: Fraction of all faults detected or detectable by BIT/ETE, Fraction of all detectable faults detected or detectable with BIT/ETE, Fraction of all faults detected through the use of defined means, Percentage of all faults automatically detected by BIT/ETE, Percentage of all faults detectable by BIT/ETE, Percentage of all faults detectable on-line by BIT/ETE, Percentage of all faults and out-of-tolerance conditions detectable by BIT/ETE, and Percentage of all faults detectable by any means.

One problem with traditional metrics is that they are "overloaded." Overloaded in this case means that due to "common understanding" of the terms, fine variations are not specified. Consequently, users of the term do not necessarily know the implications of a precise definition. Discussions of overloaded terms go on at length, in part because everyone in the discussion has brought along a lot of mental baggage. Often, progress is only made when a neutral term is chosen and the meaning is built from the ground up. This overloading is so severe, for example, that there was no definition of FFD in *System Test and Diagnosis* [2], the authors preferring to use Non-Detection Percentage (NDP). FFD is the negative of NDP and is equal to 1–NDP.

Even the number of faults counted in the field requires a more precise definition. The "overloaded" version is simply a count of all the things that failed. The quantity of all faults, as usually defined in the industry, is different. The quantity of faults detected by BIT/ETE, and the quantity of faults detected exclude the occurrence of false alarms. Intermittent faults are classified as a single fault. Temporary faults, those caused by external transients of noise, are not classified as faults.

Another aspect of the challenge is that many metrics sound different but are not. For example, Ambiguity Group Isolation Probabilities, Fault Isolation Resolution, Isolation Level, and System Operational Isolation Level mean nearly the same thing.

OTHER PRODUCT DATA APPLICATIONS

Ties to Maintenance Feedback

In 1993, a Project Authorization Request (PAR) was submitted to the IEEE for new standards project related to specifying information and services for test and maintenance information feedback. The Test and Maintenance Information Management Standard (TMIMS) project was approved by the IEEE in early 1994. The focus of this project was to define exchange and service standards (similar to AI-ESTATE) which support the test and diagnostic maturation process. In 1998, due to a lack of progress, the TMIMS PAR was cancelled. The revised AI-ESTATE models make development of the TMIMS standard achievable.

AI-ESTATE continues to require definition of exchange and service standards related to test and maintenance information. In 1998, shortly after the cancellation of the TMIMS PAR, the D&MC committee decided to include test and maintenance information in its scope. The approach will be consistent with AI-ESTATE (i.e., the definition of information models and EXPRESS-level services derived from traversing the models). The starting point for the new models will be the dynamic context model in IEEE 1232.2. By keeping track of the sequence of events during a diagnostic session, much of the historical information is identified and captured that can be used for later diagnostic maturation.

As a result of ongoing work by members of the D&MC, a proposal for a new information model addressing TMIMS issues is in preparation. The model begins with a representation of the information contained within IEEE Std 1545 [12] . This standard captures parametric test information. The TMIMS information includes parametrics, test events, maintenance events, and explicit ties to AI-ESTATE. The Dynamic Context Model defined in AI-ESTATE is forming the foundation for capturing a diagnostic session and will be the primary starting point for any connections to the historical data.

Ties to Product Descriptions

Through the 1990s, the IEEE has been developing a family of standards under the umbrella of "A Broad Based Environment for Test" (ABBET) [13,14]. An early architecture of ABBET, based on information modeling, presented ABBET as five layers: 1) product description, 2) test requirements/strategy, 3) test methods, 4) test resources, and 5) instrumentation. Since then, standards for the "lower layers" of ABBET (i.e., layers 3–5) have been defined; however, it has long been recognized that the major benefit from standardization will come from the "upper layers" (i.e., layers 1 and 2).

AI-ESTATE satisfies many of the requirements related to layer two of ABBET (however, AI-ESTATE has never been considered part of the ABBET family). Further, a recent proposal for a new information model-based standard, called the Test Requirements Model (TeRM), will address specific concerns of test requirements [15, 16]. Standards for the product description layer have always been problematic due to issues related to the revelation of intellectual property. In mid-2000, a PAR will be presented to the IEEE to cover the TeRM work. With the combination of TeRM, AI-ESTATE, and TMIMS, it is anticipated that intellectual property can be hidden from information provided in standard form while still supporting the test engineer fully.

CONCLUSION

Reasoning system technology has progressed to the point where electronic and other complex systems are employing artificial intelligence as a primary component in meeting system test and verification requirements. This is giving rise to a proliferation of AI-based design, test, and diagnostic tools. Unfortunately, the lack of standard interfaces between these reasoning systems has increased the likelihood of significantly higher product life-cycle cost. Such costs arise from redundant engineering efforts during design and test phases, sizeable investment in special-purpose tools, and loss of system configuration control.

The AI-ESTATE standards promise to facilitate ease in production testing and long-term support of systems, as well as reducing overall product life-cycle cost. This will be accomplished by facilitating portability, knowledge reuse, and sharing of test and diagnostic information among embedded, automatic, and stand-alone test systems within the broader scope of product design, manufacture, and support. AI-ESTATE was first conceived in 1988 as a standard for representing expert-system rule bases in the context of maintenance data collection. Since that time, AI-ESTATE has evolved to be embodied in three published standards related to the exchange of diagnostic information and the interaction of diagnostic reasoners within a diagnostic environment. The three standards have been recommended for inclusion on the US DoD ATS Executive Agent's list of standard satisfying requirements for ATS critical interfaces. In looking to the next generation, AI-ESTATE is expanding to address issues of testability, diagnosability, maintenance data collection, and test requirements specification.

Further information on the AI-ESTATE standards and the activities of the D&MC can be found at: http://grouper.ieee.org/groups/1232/.

ACKNOWLEDGMENTS

In many ways, it is unfortunate that a paper such as this includes only the names of two authors. The work reported here is the result of efforts of a committee of devoted volunteers who have supplied their expertise in system test and diagnosis to develop strong, sound standards supporting the diagnostics community. We would like to thank Les Orlidge, Randy Simpson, Tim Bearse, Tim Wilmering, Greg Bowman, Dave Kleinman, Lee Shombert, Sharon Goodall, Len Haynes, Jack Taylor, Amanda Giarla, Bill Simerly, and Helmut Scheibenzuber for their efforts in developing the standards and promoting their use in industry.

REFERENCES

1. MIL STD 2165. 1985. Testability Program for Electronic Systems and Equipment, Washington, DC: Naval Electronic Systems Command (ELEX-8111)

2. Simpson, W. and Sheppard, J. 1994. System Test and Diagnosis, Boston, MA: Kluwer Academic Publishers.

3. IEEE Std 1232-1995. IEEE Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Overview and Architecture, Piscataway, NJ: IEEE Standards Press.

4. IEEE Std 1232.2-1998. IEEE Trial-Use Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Service Specification, Piscataway, NJ: IEEE Standards Press.

5. IEEE Std 1232.1-1997. IEEE Trial-Use Standard for Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE): Data and Knowledge Specification, Piscataway, NJ: IEEE Standards Press.

6. ISO 10303-11:1994. Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 11: Description Methods: The EXPRESS Language Reference Manual, Geneva, Switzerland: International Organization for Standardization.

7. ISO 10303-21:1994. Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 21: Implementation Methods: Clear Text Encoding of the Exchange Structure, Geneva, Switzerland: International Organization for Standardization.

8. Schenk, D. and Wilson, P. 1994. Information Modeling: The EXPRESS Way, New York: Oxford University Press.

9. Sheppard, J. and Maguire, R. 1996. "Application Scenarios for AI-ESTATE Services," AUTOTESTCON '96 Conference Record, New York: IEEE Press.

10. Sheppard, J. and Orlidge, L. 1997. Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE)—A New Standard for System Diagnostics," Proceedings of the International Test Conference, Los Alamitos, CA: IEEE Computer Society Press.

11. Simpson, W., Bailey, J., Barto, K. and Esker, E., 1985 "Organization-Level Testability Prediction", ARINC Research Corporation Report 1511-01-3623 Prepared for the Rome Air Development Center.

12. IEEE Std 1545-1999. 1999. Standard for Parametric Data Log Format, Piscataway, NJ: IEEE Standards Press.

13. IEEE Std 1226-1993. IEEE Trial-Use Standard for A Broad Based Environment for Test (ABBET): Overview and Architecture, Piscataway, NJ: IEEE Standards Press.

14. IEEE Std 1226.6-1996. IEEE Guide to the Understanding of A Broad Based Environment for Test (ABBET), Piscataway, NJ: IEEE Standards Press.

15. Shombert, L. 1998. Test Requirements Model Language Reference Manual, Draft 0.1, Technical Report CAE-1998-07-01, Vienna, VA: Intermetrics.

16. Shombert, L. and Sheppard, J. 1998. "A Behavior Model for Next Generation Test Systems," Journal of Electronic Testing: Theory and Applications, Vol. 13, No. 3