Software Standards

Our world, the world of software, is changing every day, and the rate of change is increasing. But coupled with this rapid growth are signs that the software industry is beginning to mature.

If we examine other professions, such as accounting, law, or medicine, we can identify several characteristics of established professions that relate directly to the software industry today. First, each established profession is represented by a professional association. In software we have several, including the IEEE Computer Society and the Association for Computing Machinery. Next, each profession has a standard dictionary. In early 1983 the IEEE Standards Board approved Project 729 and issued the IEEE Standard Glossary of Software Engineering Terminology, the industry’s first consensus-based, formally balloted terminology. Third, each profession has a standard for what is considered “good practice.” Here, the software industry is still very young and growing rapidly, but numerous responses to this fast-changing area exist—for example, this department, the first in an IEEE publication dedicated solely to covering software standards.

The primary purpose of this department is to keep our readers up to date on software standards activities, including the status of present standards efforts, scheduled standards events, standards implementations, and problems with standards. We’ll begin by reviewing the need for standards at the project, multiproject, and national and international levels.

The need for software standards

Initially, software standards were created to satisfy particular needs, and the need for standard program-

Call for papers: Third Software Engineering Standards Application Workshop

Papers that summarize experience, clarify issues, or discuss the impact of the application of software engineering standards are being sought for presentation at the Third Software Engineering Standards Application Workshop, to be held October 2-4 at the Sheraton-Palace Hotel, San Francisco, Calif.

Any one of the following three approaches will qualify an applicant for participation.

- Submit a 300- to 500-word position statement.
- Submit a panel session proposal that includes a less than 300-word abstract of the topic and a list of potential participants with their addresses and affiliations.
- Submit five copies of a technical paper, not exceeding six proceedings pages (about 3000 words) and including complete author information and an abstract of 150 words or less.

Send by February 17, 1984, to M. Branstad, NBS, Rm. B266, Technology Bldg., Washington, DC 20234; (301) 921-3545. For additional information contact L. Tripp, Boeing Computer Services, PO Box 24346, Seattle, WA 98124; (206) 575-5390.
a fault (bug) takes longer. The person trying to find and eliminate the fault must know not only what is wrong with the code but what it was meant to accomplish in the first place. This requires understanding the original designer's intent and, therefore, the design technique.

It is difficult to find published statistics on the costs or savings associated with having or not having software standards in place on a software project. However, it seems correct to say that consistent, documented terminology and project standards improve communication among team members and will result in fewer misinterpretations.

At the multiproject level. Standards help reduce software development costs for multilevel organizations as well as for single-project organizations. The learning time involved in moving a person from one project to another is shorter, and therefore less costly, if both projects use the same standards. The shorter, less-expensive learning period increases software productivity at the multiproject level. The cost of hardware and software tools can be shared or spread over several projects if all the projects use the same standard software development tools.

IEEE Computer Society software standards activities

The following list, published in response to frequent questions regarding the progress of various projects, gives the status of current IEEE Computer Society software standards activities.

Standards issued. The five software standards listed below have been issued and are available from the Computer Society's West Coast Office. Send your prepaid order to IEEE-CS Order Dept., PO Box 80452, Worldway Postal Center, Los Angeles, CA 90080, giving the CS order numbers listed and adding a $2.00 shipping and handling charge.

- **Project 729, Software Engineering Terminology.** CS order number 930; nonmembers, $7.50; members, $6.75.
- **Project 730, Software Quality Assurance Plans.** CS order number 915; nonmembers, $5.00; members, $4.50.
- **Project 770, Computer Programming Language Pascal.** CS order number 917; nonmembers, $17.95; members, $16.15.
- **Project 828, Software Configuration Management Plans.** CS order number 931; nonmembers, $6.50; members, $5.85.
- **Project 829, Software Test Documentation.** CS order number 929; nonmembers, $8.00; members, $7.20.

Drafts generated. To participate in the committee's work or to receive a copy of its draft standard, contact the project chairman.

- **Project 755, High-Level Languages for Microprocessors.** This project, sponsored by the Technical Committee on Mini/Microcomputers, is chaired by Richard James III, 3705 Eastwood Cir., Santa Clara, CA 95014; (408) 988-3048.
- **Project 830, Software Requirements Specification.** Sponsored by the TC on Software Engineering, this project is chaired by A. Davis, GTE Network Systems, 2500 W. Utopia Rd., Phoenix, AZ 85027; (602) 582-7069.
- **Project 855, Microprocessor Operating System Interface.** Jack Cowan of Intel chairs Project 855, which is sponsored by the TC on Mini/Microcomputers. Information is available from the committee secretary: Sam Kirk, AMD, MS6, 3340 Scott Blvd., Santa Clara, CA 95051; (408) 988-7777.

Committee work in progress. Eight software standards projects are in various stages of committee work. Anyone interested in one of the following areas is encouraged to contact the chairman and join the working group.

- **Project 610, Computer Dictionary.** Jane Radatz chairs this group, sponsored by the Computer Standards Committee, which published an initial version of its work in 1979 (still available as CS order number 901; nonmembers, $8.00; members, $6.00). Contact Radatz at Logicom, Inc., PO Box 80158, San Diego, CA 92138; (619) 455-1330, Ext. 502.
- **Project 982, Software Reliability.** Chaired by James Dobbins, this group is sponsored by the TC on Software Engineering. Dobbins can be reached at IBM FSD, MS 105-913, 9500 Godwin Dr., Manassas, VA 22110; (703) 3677-3912.

- **Project 983, Software Quality Assurance Guide.** This project is sponsored by the TC on Software Engineering and chaired by G. Tice, Tektronix, Inc., PO Box 392, Wilsonville, OR 97070; (503) 629-1310.
- **Project 990, Ada as a Program Design Language.** This group, sponsored by the TC on Software Engineering, is chaired by Robert Blasewitz, RCA, MS 101-210, Mooresstown, NJ 08057; (609) 778-3955.
- **Project 1002, Software Engineering Taxonomy.** This project, sponsored by the TC on Software Engineering, is chaired by Leonard Tripp, MS9C-70, Boeing Computer Services, PO Box 24346, Seattle, WA 98124; (206) 575-5390.
- **Project 1003, Operating System Kernel.** Dennis Allison heads this group, which is sponsored by the TC on Operating Systems. Contact Allison at the Computer Systems Lab., Stanford University, Stanford, CA 94305; (415) 497-9213.
- **Project 1006, Programming Language C.** Information on this project, sponsored by the TC on Programming Languages, may be obtained from Michael P. Hagerty, Motorola, Inc., 3203 N. 56th St., Phoenix, AZ 85018; (602) 244-5475.
- **Project 1008, Software Unit Testing.** This project, sponsored by the TC on Software Engineering, is chaired by David Gelperin, 2245 Zealand Ave., N., Golden Valley, MN 55427; (612) 542-8620.
At the national and international level. The need for software standards, which seems straightforward at the project and multiproject levels, is less clear across company boundaries at the national and international levels.

The most obvious need is to be able to compare software development practices among companies. Today, any and all comparisons are, at best, vague. For example, the present databases on software fault discovery rates do not use the same definition for a fault. Some programmers call any problem a fault and fill out failure report forms for inclusion in the databases. Other programmers simply make corrections and continue. Therefore, when two companies report software fault find rates of faults per 1000 lines of code, they may or may not be saying the same thing.

The buyer's need to compare products is another reason for applying standards to the software development process, and a seller's adherence to IEEE/ANSI standards is a favorable comment on his product. In fact, something like the Underwriter's Laboratory seal on electrical appliances would be of tremendous value for software products, if the approval process were accepted by a large group of professionals.

The reasons for standards dealing with the software development process that we've noted are usually sufficient to start most organizations developing a set of standards. Then the questions really begin: "How will the standard affect creativity?" "What and when do we standardize?" Resolving these and other questions will require extensive discussion and effort.

Standards vs. creativity

The argument that standards restrict creativity is often heard in organizations that are dealing with new standards. Any conclusion reached is normally based upon the particular standard involved.

Music is a good example of a creative medium that has a rigorously standardized language. There are only so many notes, lines, and spaces. The mundane, simplistic features of documenting music have been thoroughly standardized, yet the creative arrangements of the notes are seemingly inexhaustible.

If software standards were applied in the same manner as music standards, the impact on creativity would be positive not negative. The creative effort in software development would be channeled toward the problems to be solved, not into areas that were standardized for that project. We could spend less time discussing what the document looks like and more time putting information in the document.

Document formats and descriptions are now typical standards, and symbols and/or syntax for describing a system's design or code are prime candidates for standardization. Such software standards, applied as suggested, would not restrict creativity.

The counter argument is that people will take a standard as an absolute and that this perception will stop, or at least slow down, the evolution of the technology area being standardized.

The opposing positions may be stated as follows:

- Standardization too early in the life of a technology may reduce its rate of development.
- Standardization too late in the life of a technology will delay the benefits of standardization.

The questions of when to standardize and what to standardize are closely related and should be answered at the national level by a consensus process. A single position selected by one author, by a small, homogeneous group, or by one company would simply reflect a limited set of conditions, not the national need.

The recent standardization of the Ada language and the ongoing effort to standardize a program design language for Ada are two good examples of software standards where debates over when and what to standardize are being resolved by a consensus process. Some opponents claim it is too early to standardize, even though Ada has been a Department of Defense standard for over a year.

In an organization, the decision to standardize may belong to one person or, perhaps, to a committee. In these cases the questions become "Which standard?" and "How do we determine whether this is a reasonable standard?" A company can best resolve these questions if national-level, consensus-based standards are available as starting points.

However, that raises specific questions about national-level standards within each development organization.

Providing the answers, directions to the answers, and a forum for ongoing pro and con discussions for software standards is the purpose of this department in IEEE Software. We look forward to the challenge.