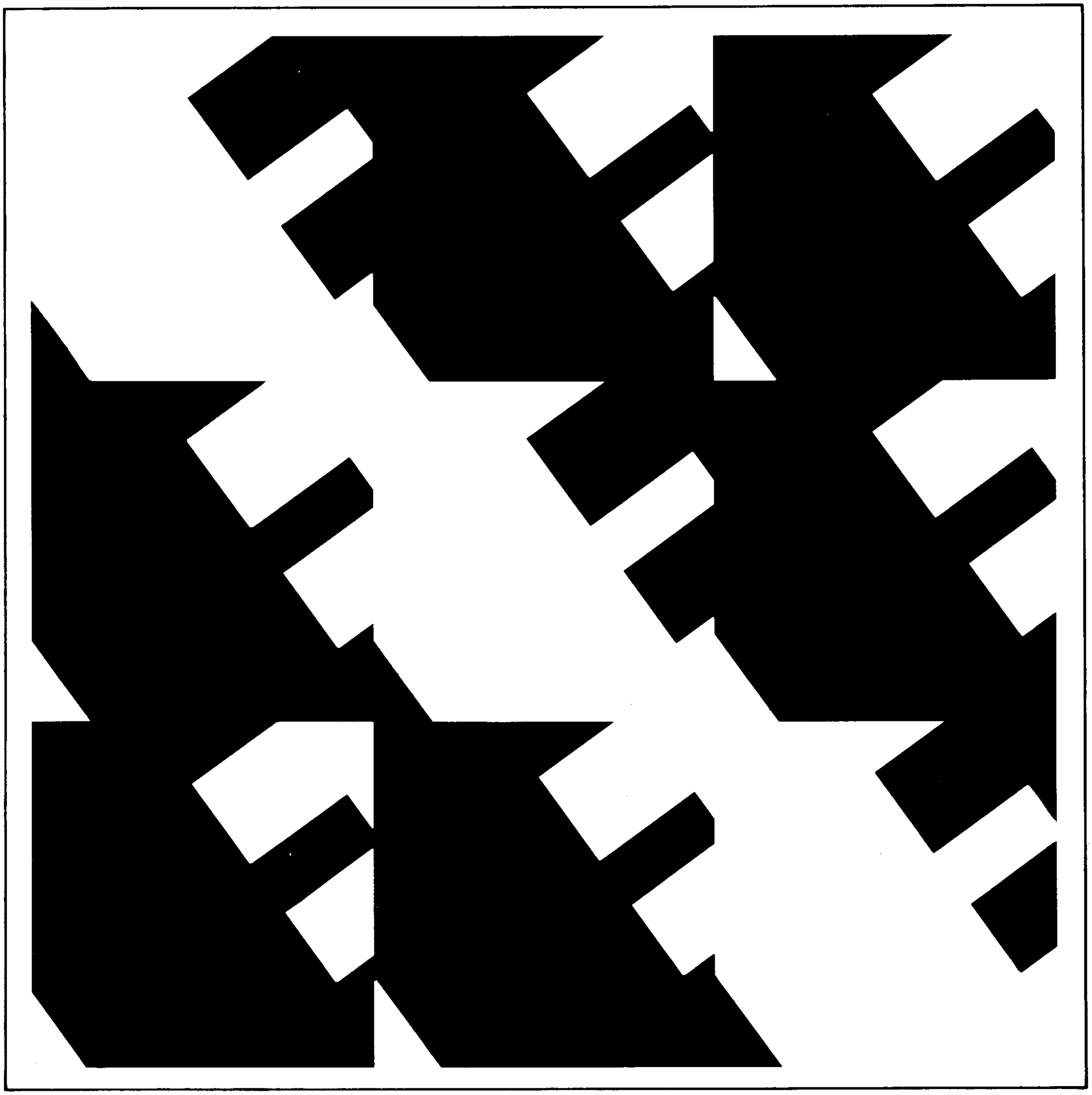
AMERICAN NATIONAL STANDARD

ANSI/IEEE Std 1002-1987

IEEE Standard Taxonomy for Software Engineering Standards



ANSI/IEEE Std 1002-1987

Published by The Institute of Electrical and Electronics Engineers, Inc 345 East 47th Street, New York, NY 10017, USA

July 20, 1987

SH11023

ANSI / IEEE
Std 1002-1987

An American National Standard

IEEE Standard Taxonomy for Software Engineering Standards

Sponsor

Software Engineering Subcommittee
of the
Technical Committee on Software Engineering
of the
IEEE Computer Society

Approved December 11, 1986

IEEE Standards Board

Approved June 4, 1987

American National Standards Institute

The Institute of Electrical and Electronics Engineers, Inc
345 East 47th Street, New York, NY 10017, USA

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE which have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least once every five years for revision or reaffirmation. When a document is more than five years old, and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
345 East 47th Street
New York, NY 10017
USA

Foreword

(This Foreword is not a part of ANSI / IEEE Std 1002-1987, IEEE Standard Taxonomy for Software Engineering Standards.)

Software Engineering is an emerging field. As part of that process a set of software engineering standards is being developed. They are used to:

(1) Improve communications between and among software engineers and others.

(2) Achieve economy of cost, human effort, and essential materials.

(3) Institutionalize practical solutions to recurring problems.

(4) Achieve predictability of cost and quality.

(5) Establish norms of acceptable professional practice.

To support the development, integration, and use of software engineering standards, a need for a taxonomy is recognized. A project was approved in June 1983 to define a taxonomy as part of a voluntary consensus process. This document is the result of that process.

This is one of an evolving set of integrated IEEE Software Engineering standards, recommended practices, and guides. The set currently includes:

ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology
ANSI/IEEE Std 730-1984, IEEE Standard for Software Quality Assurance Plans
ANSI/IEEE Std 828-1983, IEEE Standard for Software Configuration Management Plans
ANSI/IEEE Std 829-1983, IEEE Standard for Software Test Documentation
ANSI/IEEE Std 830-1984, IEEE Guide to Software Requirements Specifications
ANSI/IEEE Std 983-1986, IEEE Guide for Software Quality Assurance Planning
ANSI/IEEE Std 1008-1987, IEEE Standard for Software Unit Testing

This standard may be used in conjunction with this set of standards or separately.

The taxonomy can be applied, but is not limited to, project, program, organization, industrial, national, and international standards. As a document, this standard should be useful to those who develop, use, manage, and evaluate software engineering standards. The taxonomy provides a:

(1) Comprehensive scheme for classifying software engineering standards, recommended practices, and guides.

(2) Framework for identifying the need for new software engineering standards, recommended practices, and guides.

(3) Comprehensive scheme for analyzing a set of software engineering standards, recommended practices, and guides appropriate for a given industry, company, program, project, or particular work assignment.

(4) Framework for comparing sets of software engineering standards, recommended practices, and guides to support the selection of the most useful set for a particular software product.

The application of the taxonomy to achieve the above purposes is described in the appendix.

Keywords applicable to this standard are: nomenclature standard, notation standard, software engineering.

The sponsor for this standard was the Software Engineering Standards Subcommittee of the Software Engineering Technical Committee of the IEEE Computer Society, John W. Horch, Chairman.

Special representatives to the Software Engineering Standards Subcommittee were:

P.W. Abrahams
H.R. Berlack
A. Ferlan
S.R. Jarocki
R.R. Jones
J.A.N. Lee
J. Milandin
W.F. Mitchell
W.E. Perry
T.L. Regulinski
P.E. Schilling

The working group that developed this standard had the following membership:

Leonard L. Tripp, *Chairperson*
Perry R. Nuhn, *Co-Chairperson*
Ralph G. Wachter, *Co-Chairperson*

A. Frank Ackerman
Eleanor Antreasian
Joan P. Bateman
H. Ronald Berlack
Richard L. Chilausky
Francois Coallier
Stewart Crawford
James Darling
John W. Fendrich
Mehmet Ficici
Craig D. Fuget
David Gelperin
Jeff van Gilder
Paul Howley
John Horch
Harry Kalmbach
Louis B. Kiersky
Thomas M. Kurihara
John B. Lane
F. C. Lim
Phillip C. Marriott
Virginia Marting
Dan G. McNicholl
Mordechai Ben-Menachim
Fred Mervine
Manijeh Moghis
Dennis E. Nickle
Robert C. Olsen
Sharon R. Cobb-Pierson
Robert B. Poston
Max J. Schindler
David Schultz
Leonard W. Seagren
John Selman
David M. Siefert
Dave Simkins
R. van Tilburg
William S. Turner, III
Clyde E. Willis
Paul A. Willis

When the IEEE Standards Board approved this standard on December 11, 1986, it had the following membership:

John E. May, *Chairman*
Irving Kolodny, *Vice Chairman*
Sava I. Sherr, *Secretary*

James H. Beall
Fletcher J. Buckley
Paul G. Cummings
Donald C. Fleckenstein
Jay Forster
Daniel L. Goldberg
Kenneth D. Hendrix
Irvin N. Howell
Jack Kinn
Joseph L. Koepfinger*
Edward Lohse
Lawrence V. McCall
Donald T. Michael*
Marco W. Migliaro
Stanley Owens
John P. Riganati
Frank L. Rose
Robert E. Rountree
Martha Sloan
Oley Wanaselja
J. Richard Weger
William B. Wilkens
Helen M. Wood
Charles J. Wylie
Donald W. Zipse

*Member emeritus

The following persons were on the balloting committee that approved this document for submission to the IEEE Standards Board:

A. Frank Ackerman
Jagdish C. Agrawal
Richard L. Aurbach
K. Ramesh Babu
James Baldo, Jr
H. Jack Barnard
Roy W. Bass
Leo Beltracchi
Yechiel Ben-Naftali
Victor G. Berecz
H.R. Berlack
J. Emmett Black
Michael A. Blackledge
Ron Blair
Kevin W. Bowyer
Kathleen L. Briggs
Fletcher J. Buckley
Margaret Butler
Homer C. Carney
Ronald R. Carter
Robert N. Charette
Tsun S. Chow
Jung K. Chung
Peter Coad, Jr
Francois Coallier
Christopher M. Cooke
A.J. Cote, Jr
Stewart Crawford
George D. Darling
Taz Daughtrey
P.O. Denny
Harpal S. Dhama
Mike Dotson
William P. Dupros
Michael Dutton
Robert E. Dwyer
Mary Eads
John D. Earls
Michael Edward
L.G. Egan
Steven R. Eisen
Caroline L. Evans
David W. Favor
John W. Fendrich
Robert G. Ferreol
Glenn S. Fields
Gordon Force
J. Forster
Deborah L. Franke
C.R. Frederick
Carl Friedlander

Ismael Fuentes-Crespo
Micheel Galinier
Leonard B. Gardner
David Gelperin
James L. Gildersleeve
Shirley Gloss-Soler
Ole Golubjatnikov
J. Kaye Grau
Andrej Grebenc
Thomas Griest
Victor M. Guarnera
Lawrence M. Gunther
David A. Gustafson
Russell Gustin
Howard Hamer
Hans-Ludwig Hansen
George B. Hawthorne
Clark M. Hay
Terry L. Hengl
Charles P. Hollocker
John W. Horch
Cheng Hu
Shang-Sheng Jeng
David Johnson, III
Laurel V. Kaleda
Constantine Kaniklidis
Myron S. Karasik
Adi N. Kasad
Ron Kenett
R.A. Kessler
Joseph A. Krupinski
Hirayr M. Kudyan
Joan Kundig
T.M. Kurihara
Robin B. Lake
Lak-Ming Lam
John B. Lane
Robert A.C. Lane
William P. LaPlant, Jr
Greg Larsen
Jack A. Latimer
Jay Levenson
Leon S. Levy
Paul Lieberaz
F.C. Lim
Bertil Lindberg
David Linsson
William M. Lively
John M. Long
John Lowell
L.J. Mablack

Bill Macre
Andy Mahindru
Henry Malec
Paulo Cesar Marcondes
Stuart Marcotte
Philip C. Marriott
Nicholas L. Marselos
Roger J. Martin
Robert F. Martini
Ivano Mazza
J.A. McCall
John McKissick, Jr
Glen A. Meldrum
Belden Menkus
Ben W. Miller
Manijeh Moghis
Charles S. Mooney
Joyce E. Mortison
Gene T. Morun
Dale N. Murray
Myron L. Nack
Hironobu Nagano
Saied Najafi
Gerry Neidhart
Brian Nejmeh
Dennis E. Nickle
I.H. Obbink
Wilma Osborne
D.J. Ostrom
David G. Owens
Thomas D. Parrish
M.T. Perkins
Donald J. Pfeiffer
R.M. Poston
Peter Ron Prinzivalli
Thomas S. Radi
Meir Razy
John Reddan
Larry K. Reed
Matthias F. Reese, III
T.D. Regulinski
Donald J. Reifer
Steven M. Rowan
R. Waldo Roth
Hom Sack
Julio Gonzalez Sanz
Stephen R. Schach
Franz P. Schauer
Max Schindler
Norman Schneidewind
Nolf A. Schnoege

Robert G. Schueppert
David J. Schultz
Gregory D. Schumacher
Leonard W. Seagren
Devdoot Sen
Gerard P. Share
Robert W. Shillato
David M. Siefert
David J. Simkins
Jacob Slonim
Marion P. Smith
Harry M. Sneed
J.G. Snodgrass
Al R. Sorkowitz
Hugh B. Spillane
Lee Sprague
G. Wayne Staley
Alan N. Sukert
William G. Sutcliffe
Richard H. Thayer
Booker T. Thomas
Paul U. Thompson
E.O. Tilford
Terrence L. Tillmanns
Lawrence F. Tracey
Glendon R. Trebble
Robert Troy
C.L. Troyanowski
Dana L. Ulery
R.L. Van Tilburg
P.M. Vater
Osmo Vikman
R. Wachter
Dolores R. Wallace
Andrew H. Weigel
R.W. Werlwas
Walter L. Whipple
Paul A. Willis
Patrick J. Wilson
Paul Wolfgang
W. Martin Wong
Dennis L. Wood
Paul R. Work
Tom Worthington
Charles Wortz
Stephen D. Yaste
Natalie C. Yopconka
Michael E. York
Marvin Zelkowitz
Peter F. Zoll

Contents

SECTION PAGE

1. Introduction 9
   1.1 Scope 9
   1.2 Terminology 9
   1.3 References 9

2. Definitions 9

3. Taxonomy of Software Engineering Standards 11
   3.1 Standards Partition 11
   3.2 Software Engineering Partition 11
   3.3 Taxonomy Framework 12

FIGURES

Fig 1 Partition of Standards by Type 11
Fig 2 Partition of Software Engineering by Function and Life Cycle 12
Fig 3 Basic Taxonomy Framework (Version A) 13
Fig 4 Basic Taxomony Framework (Version B) 14
Fig 5 Comprehensive Taxonomy Framework 15

APPENDIXES

Taxonomy Usage Examples 17

APPENDIX FIGURES

Fig A1 Example of General Standards Classification (Phase Independent) 19
Fig A2 Example of General Standards Classification (Design Phase) 20
Fig A3 Classification of IEEE Software Engineering Standards (Gross Level) 23
Fig A4 Classification of IEEE Software Engineering Standards (Refined Level-Part I) 24
Fig A5 Classification of IEEE Software Engineering Standards (Refined Level-Part II) 25
Fig A6 Job Function–Software Life Cycle Correlation 26

An American National Standard

IEEE Standard Taxonomy for Software Engineering Standards

1. Introduction

1.1 Scope. This document describes the form and content of a software engineering standards taxonomy. Applicability is not restricted by software application, size, complexity, criticality, or hardware environment. This taxonomy applies to standards (from the related disciplines of engineering management, systems engineering, computer hardware engineering, computer science, and information science) with which a software engineer would be reasonably acquainted. This taxonomy is application independent. For example, an accounting test standard would be placed under test standards, but the qualifier, accounting, has no significance. The document explains the various types of software engineering standards, their functional and external relationships, and the role of various functions participating in the software life cycle. The taxonomy may be used as a method for planning the development or evaluation of standards for an organization. It could also serve as a basis for classifying a set of standards or for organizing a standards manual.

1.2 Terminology. The word *shall* identifies the mandatory material within this standard. The words *should* and *may* identify optional material.

1.3 References. This standard shall be used in conjunction with the following reference:

[1] ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology.[1]

2. Definitions

The definitions listed below establish meaning in the context of this standard. Other definitions can be found in ANSI/IEEE Std 729-1983 [1].[2] **See specifically: audit, certification, configuration management, conversion, debugging, design, design phase, implementation phase, installation and checkout phase, integration, maintenance, operation and maintenance phase, quality assurance, requirements analysis, requirements phase, retirement phase, review, software engineering, software maintenance, test phase, and testing.** For the purpose of this standard, the term "software" includes the computer programs, data, and documentation portions of both software and firmware.

code of ethics standard. A standard that describes the characteristics of a set of moral principles dealing with accepted standards of conduct by, within, and among professions.

coding. The transforming of logic and data from design specifications into a programming language.

component standard. A standard that describes the characteristics of data or program components.

concept phase. The period of time in the software life cycle during which the user needs are described and evaluated through documentation (for example, statement of needs, advance planning report, project initiation memo, feasibility studies, system definition documentation, regulations, procedures or policies relevant to the project).

[1] ANSI/IEEE publications can be obtained from the Sales Department, American National Standards Institute, 1430 Broadway, New York, NY 10018, or from the Service Center, The Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331.

[2] The numbers in square brackets refer to those of the references listed in 1.3.

curriculum standard. A standard that describes the characteristics of a course of study on a body of knowledge that is offered by an educational institution.

description standard. A standard that describes the characteristics of product information or procedures provided to help understand, test, install, operate, or maintain the product.

design standard. A standard that describes the characteristics of a design or a design description of data or program components.

job function. A group of engineering processes that is identified as a unit for the purposes of work organization, assignment, or evaluation. Examples are design, testing, or configuration management.

language standard. A standard that describes the characteristics of a language used to describe a requirements specification, a design, or test data.

licensing standard. A standard that describes the characteristics of an authorization given by an official or a legal authority to an individual or organization to do or own a specified thing.

manufacturing phase. The period of time in the software life cycle during which the basic version of a software product is adapted to a specified set of operational environments and is distributed to a customer base.

measurement standard. A standard that describes the characteristics of evaluating a process or product.

method standard. A standard that describes the characteristics of the orderly process or procedure used in the engineering of a product or performing a service.

nomenclature standard. A standard that describes the characteristics of a system or set of names, or designations, or symbols.

notation standard. A standard that describes the characteristics of formal interchanges within a profession.

occupational title standard. A standard that describes the characteristics of the general area of work or profession.

plan standard. A standard that describes the characteristics of a scheme for accomplishing defined objectives or work within specified resources.

process management. The direction, control, and coordination of work performed to develop a product or perform a service. Example is quality assurance.

process standard. A standard that deals with the series of actions or operations used in making or achieving a product.

product analysis. The process of evaluating a product by manual or automated means to determine if the product has certain characteristics.

product engineering. The technical processes to define, design, and construct or assemble a product.

product management. The definition, coordination, and control of the characteristics of a product during its development cycle. Example is configuration management.

product standard. A standard that defines what constitutes completeness and acceptability of items that are used or produced, formally or informally, during the software engineering process.

product support. The providing of information, assistance, and training to install and make software operational in its intended environment and to distribute improved capabilities to users.

professional standard. A standard that identifies a profession as a discipline and distinguishes it from other professions.

report standard. A standard that describes the characteristics of describing results of engineering and management activities.

representation standard. A standard that describes the characteristics of portraying aspects of an engineering or management product.

requirement standard. A standard that describes the characteristics of a requirements specification.

resource management. The identification, estimation, allocation, and monitoring of the means used to develop a product or perform a service. Example is estimating.

software life cycle. The period of time that starts when a software product is conceived and ends when the product is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, manufacturing phase, installation and checkout phase, operation and maintenance phase, and sometimes, retirement phase.

taxonomy. A scheme that partitions a body of knowledge and defines the relationships among the pieces. It is used for classifying and understanding the body of knowledge.

technical management. The application of technical and administrative resources to plan, organize, and control engineering functions.

technique standard. A standard that describes the characteristics of applying accumulated technical or management skills and methods in the creation of a product or performing a service.

verification and validation. The process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements.

3. Taxonomy of Software Engineering Standards

The taxonomy shall consist of a standards partition, software engineering partition, and a framework that relates the two partitions. Each partition results in the definition of a set of categories wherein each category has a name and a membership rule. The standards partition characterizes the roles of standards. The software engineering partition characterizes the aspects of software engineering with which a standard can be associated. The framework combines the two partitions into a two-dimensional scheme, which describes the set of possible software engineering standards. The taxonomy framework also describes how the categories are organized for classification purposes. Section 3.1 describes the standards partition, Section 3.2 describes the software engineering partition, and Section 3.3 describes the taxonomy framework and its relationships.

3.1 Standards Partition. The standards partition shall be organized by type of standard. The four types are process, product, professional, and notation standards. See Fig 1 for the complete partition.

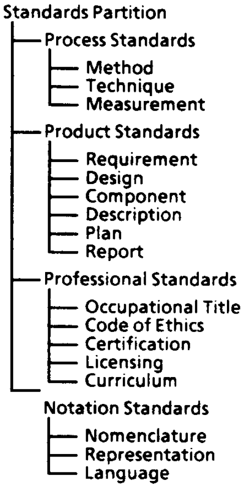
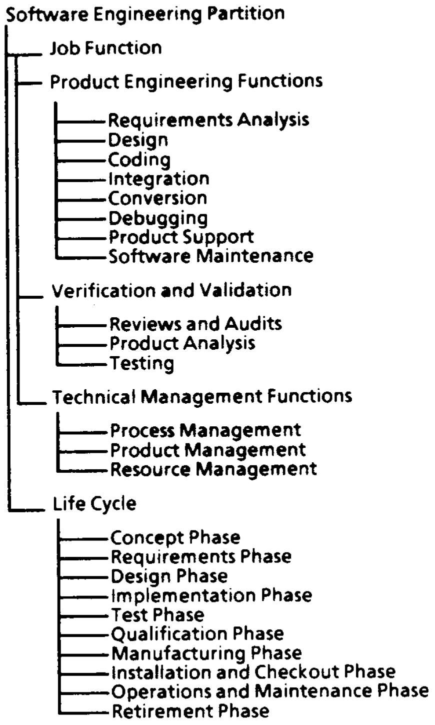


Fig 1
Partition of Standards by Type

Process standards deal with the series of actions or operations used in engineering a product or delivering a service. The actions or operations make use of methods, tools, and techniques. They give the "whos," "whats," "hows," "wheres," "whens," and levels of the work done in software engineering. Product standards are concerned with the format and content of things. The products are the documented results of the software development and maintenance activities and provide a baseline for future activities. Professional standards deal with all aspects of software engineering that identify it as a profession. An example is a curriculum for a Master of Software Engineering degree. Notation standards deal with the communication of common items among the software engineering professionals in a uniform manner. An example is a glossary. The output of a process is a product; the process is performed by people using tools and techniques within the profession.

3.2 Software Engineering Partition. The software engineering partition shall consist of two parts: job functions and software life cycle. These two parts or perspectives are used in order to compare, judge, evaluate, and determine the scope and content of software engineering standards. See Fig 2 for the software engineering



- Software Engineering Partition
  - Job Function
    - Product Engineering Functions
      - Requirements Analysis
      - Design
      - Coding
      - Integration
      - Conversion
      - Debugging
      - Product Support
      - Software Maintenance
    - Verification and Validation
      - Reviews and Audits
      - Product Analysis
      - Testing
    - Technical Management Functions
      - Process Management
      - Product Management
      - Resource Management
  - Life Cycle
    - Concept Phase
    - Requirements Phase
    - Design Phase
    - Implementation Phase
    - Test Phase
    - Qualification Phase
    - Manufacturing Phase
    - Installation and Checkout Phase
    - Operations and Maintenance Phase
    - Retirement Phase

Fig 2
Partition of Software Engineering by Function and Life Cycle

partition. Job functions are the identifiable processes of software engineering. Job functions often occur in parallel. For example, designs are updated as software elements are developed. No strict temporal sequence exists among the job functions since planning, execution, or follow-up within a function will certainly overlap other job functions.

Job functions are divided into three parts: product engineering functions, verification and validation functions, and technical management functions. The three parts contain the major ongoing, parallel activities of producing, checking, and controlling that are not concentrated in a single life cycle phase. The product engineering functions includes those processes that are necessary to define, produce, and support the final software product. Verification and validation functions are the technical activities that check the quality of the product. Technical management functions are those processes that structure and control the engineering functions. Project management is viewed as being related to technical management in the following way: Typically, project management is the use, by one or more organizations, of the technical management functions of process management, product management, and resource management to develop a product within specified resources.

3.3 Taxonomy Framework. The taxonomy framework shall consist of:

(1) Names of the categories in the standards partition and the relationships among the names

(2) Names of the categories in the software engineering partition and the relationships among the names

(3) Rules for composing the framework

(4) Presentation format for the framework

The taxonomy may be presented in different ways, depending on how it can be used most effectively. The rows and columns may be reversed, higher or lower levels of classification can be shown, or only part of the table may be used.

This standard presents three versions of the taxonomy framework for use. The three versions are titled:

(a) Basic Taxonomy Framework (Version A)

(b) Basic Taxonomy Framework (Version B)

(c) Comprehensive Taxonomy Framework

The two Basic Taxonomy Frameworks have the same column labels with the row labels being somewhat different. The row labels for Version A are a selection from the job function portion of the software engineering partition that generally are present in all software life phases and the software life cycle phases. The column labels are the major categories of the standards partition. The row labels for Version B are the complete job function portion of the software engineering partition.

The two Basic Taxonomy Frameworks are illustrated in Figs 3 and 4. The frameworks are presented in the form of a two-dimensional table. An entry in one of the tables is defined by the names from the respective row label and column label of the entry. For example, in Fig 4, the most upper left table entry would be process standards for requirements analysis.

The Comprehensive Taxonomy Framework (see Fig 5) uses the full depth of both the standards partition and the software engineering partition. For presentation purposes, the framework is organized into two parts with the row labels from the standards partition and the column labels from the software engineering partition. For this framework, the entry name is defined by the names of the respective column label and row label of the entry.

The framework composition rules define the layout for the framework and how the entries in the table are composed. The rules are:

(1) The framework is displayed as a two-dimensional table with a set of labels for the rows and a set of labels for the columns.

(2) The names from either the standards partition or the software engineering partition are assigned as the source for the row labels. The remaining partition is the source for the column labels.

(3) A suitable set of names for the row and column labels is selected from the lists shown in Figs 1 and 2, starting at the left and proceeding to the desired level of detail.

(4) The scope of the framework is defined by eliminating those row-column pairs that are not feasible.

(5) An entry in the table is defined by names from the respective row and column of the entry.

Examples of how to classify standards using this taxonomy are contained in Appendix A.

			Type of Standard			
			Process Standard	Product Standard	Professional Standard	Notation Standard
Job Function	Ver & Val	Reviews & Audits				
		Product Analysis				
		Testing				
	Tech Mgmt	Process Management				
		Product Management				
		Resource Management				
S/W Life Cycle	Concept					
	Requirement					
	Design					
	Implementation					
	Test					
	Manufacturing					
	Operation and Maintenance					
	Retirement					

Fig 3
Basic Taxonomy Framework (Version A)

			Type of Standard			
			Process Standard	Product Standard	Professional Standard	Notation Standard
Job Function	Product Engineering	Requirements Analysis				
		Design				
		Coding				
		Integration				
		Conversion				
		Debugging				
		Product Support				
		Software Maintenance				
	Ver & Val	Reviews and Audits				
		Product Analysis				
		Testing				
	Tech Mgmt	Process Management				
		Product Management				
		Resource Management				

Fig 4
Basic Taxonomy Framework (Version B)

			Job Function													
			Product Engineering								Verification & Validation			Technical Management		
			Requirements Analysis	Design	Coding	Integration	Conversion	Debugging	Product Support	Software Maintaince	Reviews and Audits	Product Analysis	Testing	Process Management	Product Management	Resource Management
Type Of Standard	Process	Method														
		Technique														
		Measurement														
	Product	Requirements														
		Design														
		Component														
		Description														
		Plan														
		Report														
	Profession	Occupational Title														
		Code of Ethics														
		Certification														
		Licensing														
		Curriculum														
	Notation	Nomenclature														
		Representation														
		Language														

Fig 5
Comprehensive Taxonomy Framework
(Part 1)

			Software Life Cycle								
			Concept	Requirements	Design	Implementation	Test	Manufacturing	Installation & Checkout	Operation & Maintenance	Retirement
Type Of Standard	Process	Method									
		Technique									
		Measurement									
	Product	Requirements									
		Design									
		Component									
		Description									
		Plan									
		Report									
	Profession	Occupational Title									
		Code of Ethics									
		Certification									
		Licensing									
		Curriculum									
	Notation	Nomenclature									
		Representation									
		Language									

**Fig 5 (Cont'd)
Comprehensive Taxonomy Framework
(Part 2)**

Appendix
Taxonomy Usage Examples

(This Appendix is not a part of ANSI / IEEE Std 1002-1987, IEEE Standard Taxonomy for Software Engineering Standards, but is included for information only.)

This Appendix illustrates how the taxonomy can be used to:

(1) Classify a set of software engineering standards

(2) Annotate software engineering standards with keywords

(3) Characterize a software engineering standards program

(4) Correlate functions and software life cycle viewpoints

A1. Classification of Selected Standards

This section presents a selection of references on software engineering standards. The key for selection was that the reference is publicly available through a trade association, government agency, or national society other than IEEE. The references are listed below with their identifier. The identifiers are placed in the two tables (Figs A1 and A2). The selected standards were classified using the job function table of the Comprehensive Taxonomy Framework organized by software life cycle phase. In a complete example, there would be a job function table for each software life cycle phase. The example presented contains two tables. The first table (see Fig A1) depicts those standards that essentially have equal applicability over most software life cycle phases. The second table (see Fig A2) depicts those standards that are of special importance for the design phase of the software life cycle.

Identifier	*Title*
ICAM	Air Force Materials Laboratory, ICAM Documentation Standards, IDS 150120000A, December 28, 1981.
480	Department of Defense, Configuration Control-Engineering Changes, Deviations, and Waivers, DOD-STD-480A, 1978.[3]
483	Department of Defense, Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs, MIL-STD-483A, June 4, 1985.[4]
499	Department of Defense, Engineering Management, MIL-STD-499, May 1, 1974.
52779	Department of Defense, Software Quality Assurance Program Requirements, MIL-S-52779A, August 1, 1979.
490	Department of Defense, Specification Practices, MIL-STD-490, June 4, 1985.
RADC	Rome Air Development Center, RADC Computer Software Development Specification, CP 0787796100E, May 1979.
TADSTAD9	Department of Defense, Tactical Digital System Standard, Software Quality Assurance Testing Criteria, TADSTAD 9, 1978.[5]
1521	Department of Defense, Technical Reviews and Audits for Systems, Equipment, and Computer Software, MIL-STD-1521B, June 4, 1985.
2167	Department of Defense, Defense System Software Department, DOD-STD-2167, June 4, 1985.

[3] DOD and MIL publications are available from the Director, US Navy Publications and Printing Service, Eastern Division, 700 Robbins Avenue, Philadelphia, PA 19111.

[4] See footnote 3.

[5] Information on this publication can be obtained by writing to TAD, Chief of Materiel Command Headquarters, Washington, DC 20360.

Identifier	*Title*	
2167.1	Section 5.1	Requirements Analysis
2167.2	Sections 5.2, 5.3	Design
2167.3	Section 5.4	Coding
2167.4	Sections 5.5, 5.6	Integration and Testing
2167.5	Section 5.7	Configuration Management
2167.6	Section 5.8	Quality Evaluation
2167.7	Section 5.8.1.5	Installation and Checkout
2167.8	Sections 4.1, 4.2, 5.9	Project Management
FIPS 38	National Bureau of Standards, Guidelines for Documentation of Computer Programs and Automated Data Systems, Federal Information Processing Standards (FIPS) Publication (PUB) 38, February 15, 1976.[6]	
FIPS 64	National Bureau of Standards, Guidelines for Documentation of Computer Programs and Automated Data Systems for the Initiation Phase, FIPS PUB 64, August 1, 1979.	
FIPS 99	National Bureau of Standards, Guideline: A Framework for the Evaluation and Comparison of Software Development Tools, FIPS PUB 99, March 1983.	
FIPS 101	National Bureau of Standards, Guideline for Lifecycle Validation, Verification, and Testing of Computer Software, FIPS PUB 101, June 1983.	
FIPS 105	National Bureau of Standards, Guideline for Software Documentation Management, FIPS PUB 105, June 1984.	
FIPS 106	National Bureau of Standards, Guideline on Software Maintenance, FIPS PUB 106, July 1984.	
NSAC-39	Nuclear Safety Analysis Center, Verification and Validation for Safety Parameter Display Systems, NSAC-39, December 1981.	
178	Radio Technical Commission for Aeronautics, Software Considerations in Airborne Systems and Equipment Certification, RTCA/DO-178A, March 22, 1985.[7]	
178.1	Section 6	Development Verification and Validation
178.2	Sections 7.1, 7.2	Configuration Management
178.3	Sections 7.1, 7.3	Software Quality Assurance
9650	MITRE, Software Reporting Metrics, ESD-TR-85-145, MTR 9650, Revision 2, November 1985.	

[6] FIPS publications are available from the Standards Processing Coordinator, Institute for Computer Sciences and Technology, National Bureau of Standards, Gaithersburg, MD 20899.

[7] RTCA publications are available from the Radio Technical Commission for Aeronautics (RTCA), 1425 K Street, NW, Suite 500, Washington, DC 20005.

			Type of Standard			
			Process Standard	Product Standard	Professional Standard	Notation Standard
Job Functions	Product Engineering	Requirement Analysis	FIPS 99			
		Design	FIPS 99			
		Coding	FIPS 99			
		Integration	FIPS 99			
		Conversion	FIPS 99			
		Debugging	FIPS 99			
		Product Support	FIPS 99			
		Software Maintenance	FIPS 99			
	V & V	Reviews and Audits	1521,178.1, FIPS 101 NSAC-39			
		Product Analysis	178.1, FIPS 101 NSAC-39			
		Testing	TADSTAD 9, 178.1, FIPS 101, NSAC-39			
	Tech Mgmt	Process Management	52779, 2167.6, 1521, FIPS 105, RADC, 2167.8,178.3	2167		
		Product Management	480, 483, 178.2, 2167.5	483, 2167		
		Resource Management	2167.8, 9650			

Fig A1
Example of General Standard Classification (Phase Independent)

			Type of Standard			
			Process Standard	Product Standard	Professional Standard	Notation Standard
Job Functions	Product Engineering	Requirement Analysis	499*, 2167.1*	2167.1*, FIPS 64*, ICAM*		
		Design	2167.2, RADC	2167.2, FIPS 38, ICAM, 490		
		Coding	2167.3*, RADC*	2167.3*, ICAM*		
		Integration	2167.4	2167.4		
		Conversion				
		Debugging				
		Product Support				
		Software Maintenance	FIPS 106*			
	V & V	Reviews and Audits	1521, 178.1, FIPS 101, NSAC-39			
		Product Analysis	178.1, FIPS 101 NSAC-39			
		Testing	TADSTAD 9, 178.1, FIPS 101, NSAC-39			
	Tech Mgmt	Process Management	52779, 2167.6, 1521, FIPS 105, RADC, 2167.8,178.3	2167		
		Product Management	480, 483, 178.2, 2167.5	483, 2167		
		Resource Management	2167.8, 9650			

Legend *Examine for planning purposes

Fig A2
Example of General Standard Classification (Design Phase)

A2. An Approach to Annotating Software Engineering Standards with Keywords

The process of analysis, selection, and comparing of standards will benefit from a systematic means of keyword identification, which may then be incorporated into an organization's classification and retrieval procedures. An example set of keyword formation rules follows:

(1) Software engineering standards shall be classified with keywords. This shall be accomplished as part of a standard's development.

(2) Keywords shall be included in a standard's introduction. Keyword inclusion shall use the following format: "Keywords applicable to this standard are: Keyword 1, Keyword 2, . . ., Keyword n."

(3) Keywords shall be limited to words or phrases as contained in IEEE Std 1002-1987.

(4) Multiple keywords may be used in classifying a standard.

(5) Commas shall be used to separate keywords. The keyword shall will be terminated with a period.

(6) A standard shall be assigned at least one keyword from both the standards partition and software engineering partition. Within the categories of function and life cycle, multiple primary keywords may be selected.

The application of the keyword rules to some of the IEEE software engineering standards is illustrated in the following list:

Example #1. ANSI/IEEE Std 729-1983, IEEE Standard Glossary of Software Engineering Terminology. Keywords applicable to this standard are: nomenclature standard, notation standard, software engineering.

Example #2. ANSI/IEEE Std 730-1984, IEEE Standard for Software Quality Assurance Plans. Keywords applicable to this standard are: process management, product standard, software engineering, technical management.

Example #3. ANSI/IEEE Std 828-1983, IEEE Standard for Software Configuration Management Plans. Keywords applicable to this standard are: product management, product standard, technical management, software engineering.

Example #4. ANSI/IEEE Std 829-1983, IEEE Standard for Software Test Documentation. Keywords applicable to this standard are: product standard, software engineering, testing, verification and validation.

Example #5. ANSI/IEEE Std 830-1984, IEEE Guide to Software Requirements Specifications. Keywords applicable to this standard are: product engineering, product standard, requirements analysis, software engineering.

Example #6. ANSI/IEEE Std 983-1986, IEEE Guide to Software Quality Assurance Planning. Keywords applicable to this standard are: process standard, process management, technical management, software engineering.

Example #7. ANSI/IEEE Std 1008-1987, IEEE Standard for Software Unit Testing. Keywords applicable to this standard are: process standard, testing, verification and validation, software engineering.

A3. Application of Taxonomy to IEEE Software Engineering Standards (SES) Program

The IEEE Technical Committee on Software Engineering has an active program for software engineering standards. Listed below are the standards that are complete and those that are still in progress. The list of standards has been categorized by the taxonomy. To do that, three tables were created. The first table (see Fig A3) consists of the job function portion of the software engineering partition down the side and standards partition across the top. This orientation was chosen for presentation purposes.

Each entry on the standards list below was placed in the appropriate table entry. The S, R, and G refer to standard, recommended practice, and guide, respectively. The empty entries indicate possible areas for future standards.

The second and third tables use the standards partition down the side and functions across the top. The next lower level of detail was added for the standards partition. See Figs A4 and A5.

Approved Software Engineering Standards

Ref	*Description*
729	IEEE Standard Glossary or Software Engineering Terminology
730	IEEE Standard for Software Quality Assurance Plans
828	IEEE Standard for Software Configuration Management Plans

Ref	*Description*
829	IEEE Standard for Software Test Documentation
830	IEEE Guide to Software Requirements Specifications
983	IEEE Guide for Software Quality Assurance Planning
990	IEEE Guide for the Use of Ada* As a PDL
1002	IEEE Standard Taxonomy for Software Engineering Standards
1008	IEEE Standard for Software Unit Testing
1012	IEEE Standard for Software Verification and Validation Plans
1016	IEEE Recommended Practice for Software Design Descriptions

Approved Software Engineering Standards Projects

Ref	*Description*
P982	Standard for Software Reliability Measurement
P1028	Standard for Software Reviews and Audits
P1042	Guide for Software Configuration Management
P1044	Standard Classification of Software Errors, Faults, and Failures
P1045	Standard for Software Productivity Metrics
P1058	Standard for the Software Project Management Plan
P1059	Guide for Software Verification and Validation
P1060	Standard for Software Maintenance
P1061	Standard for Software Quality Metrics
P1062	Guide for Third Party Software Acquisition
P1063	Standard for User Documentation
P1074	Standard for the Software Life Cycle Processes

A4. Job Function to Software Life Cycle Correlation

In some sense, job functions and phases can be correlated to each other. The purpose of this section is to illustrate that relationship. See Fig A6.

Note that in the product engineering and verification and validation categories each row is filled in to indicate where

(1) the planning or monitoring activity takes place (empty square)

(2) the focus of the phase and job function partially coincide (shaded square)

(3) the focus of the phase and job function directly coincide (dark square)

For product engineering and verification and validation activities, this indicates the respective phases for which these activities build, reach and stay at peak effort, and then taper off. The maintenance phase is typically a repeat of the basic software life cycle, and this is denoted in the respective column by an asterisk.

Note that for the technical management functions, activities generally happen across all phases. This is indicated by dark squares for all phases for these job functions.

* Ada is a registered trademark of the U.S. Government, AJPO.

			Type of Standard			
			Process Standard	Product Standard	Professional Standard	Notation Standard
Job Functions	Product Engineering	Requirement Analysis	1074(S)	830(G)		729, 1002(S)
		Design	1074(S)	1016(R)		729(S), 990(R), 1002(S), 1016(R)
		Coding	1074(S)			729(S), 1002(S)
		Integration	1074(S)			729(S), 1002(S)
		Conversion	1074(S)			729(S), 1002(S)
		Debugging	1074(S)			729(S), 1002(S)
		Product Support	1074(S)	1063(S)		729(S), 1002(S)
		Software Maintenance	1060(S), 1074(S)			729(S), 1002(S)
	V & V	Review and Audits	1028(S), 1074(S)	1012(S)		729(S), 1002(S)
		Product Analysis	1059(G)			729(S), 1002(S)
		Testing	829(S), 1008(S), 1012(S), 1074(S), 1059(G)	829(S), 1012(S)		729(S), 1002(S)
	Tech Mgmt	Process Management	1028(S), 1062(S), 1074(S), 983(G), 1061(S)	730(S), 1058(S)		729(S), 1002(S)
		Product Management	982(S), 1028(S), 1042(G), 1044(S), 1074(S)	828(S), 1063(S)		729(S), 1002(S)
		Resource Management	1045(S)			729(S), 1002(S)

Fig A3
Classification of IEEE Software Engineering Standards (Gross Level)

			Job Function							
			Product Engineering							
			Requirements Analysis	Design	Coding	Integration	Conversion	Debugging	Software Maintenance	Product Support
Type of Standard	Process	Method	1074	1074	1074	1074	1074	1074	1060, 1074	1074
		Technique								
		Measurement								
	Product	Requirement	830							
		Design		1016						
		Component								
		Description								1063
		Plan								
		Report								
	Profession	Occupational Title								
		Code of Ethics								
		Certification								
		Licensing								
		Curriculum								
	Notat	Nomenclature	729,1002	729,1002	729,1002	729,1002	729,1002	729,1002	729,1002	729,1002
		Representation		1016						
		Language		990						

Fig A4
Classification of IEEE Software Engineering Standards (Refined Level—Part I)

			Job Function					
			Technical Management Functions			Verification & Validation		
			Process Management	Product Management	Resource Management	Review and Audits	Product Analysis	Testing
Type of Standard	Process	Method	983,1028,1062,1074	1028,1042,1074	1074	1028, 1074	1074	829, 1008, 1074
		Technique						
		Measurement	1061	982, 1044	1045			
	Product	Requirement						
		Design						
		Component						
		Description		1063				829
		Plan	730, 1058	828		1012	1012	829, 1012
		Report						829
	Profession	Occupational						
		Code of Ethics						
		Certification						
		Licensing						
		Curriculum						
	Notat	Nomenclature	729,1002	729,1002	729,1002	729,1002	729,1002	729,1002
		Representation						
		Language						

Fig A5
Classification of IEEE Software Engineering Standards (Refined Level—Part II)

			Software Life Cycle								
			Concept	Requirements Definition	Design	Implementation	Test	Manufacturing	Installation & Checkout	Operation & Maintenance	Retirement
Job Function	Product Engineering	Requirements Analysis	■	■	▨	□	□	■	□	*	
		Design	▨	▨	■	▨	□	▨	□	*	
		Coding	□	□	▨	■	▨	▨	□	*	
		Integration	□	▨	▨	▨	■	■	□	*	
		Conversion	□	▨	▨	▨	▨	▨	■	*	
		Debugging	□	□	▨	■	■	▨	▨	*	
		Product Support	□	▨	▨	▨	▨	▨	■	■	■
		Software Maintenance	□	□	□	□	□	▨	▨	■	▨
	V & V	Review and Audits	▨	■	■	■	■	■	■	*	□
		Product Analysis	□	■	■	■	▨	▨	▨	*	
		Testing	□	□	▨	▨	■	■	▨	*	
	Tech Mgmt	Process Management	■	■	■	■	■	■	■	■	□
		Product Management	▨	■	■	■	■	■	■	■	□
		Resource Management	■	■	■	■	■	■	■	■	□

Legend:

■	Primary Role
▨	Support Role
□	Planning/Monitoring role
*	Repeat of Life Cycle

Fig A6
Job Function–Software Life Cycle Correlation

Acknowledgements

The following organizations provided support for the development of this standard:

AccuRay Corporation
Applied Physics Laboratory
AT&T Bell
Canada Bell
Northern Research
The Boeing Company
Bradley University
Computer Sciences Corporation
E-Systems
Edinboro University of Pennsylvania
Hewlett-Packard
Hughes
IBM
INCO, Inc.
ITT Corporation
McDonnell Douglas
Mervine and Pallesen
MIV-MEDA Ltd.
NCR Corporation
Northern Telecom
Pratt & Whitney Aircraft
Programming
Environments, Inc.
Sanders Associates Software
Engineering Associates Software
Quality Engineering
Teledyne Brown Engineering
Tennessee Valley Authority
The Algoma Steel Corporation, Ltd.
U.S. Department of Housing and Urban Development
U.S. Department of Transportation

This support does not constitute or imply approval or endorsement of this standard.