TECHNICAL REPORT

ISO/IEC TR 12182

First edition
1998-11-01

Information technology — Categorization of software

Technologies de l'information — Classement des logiciels

Reference number
ISO/IEC TR 12182:1998(E)

Contents

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The main task of technical committees is to prepare International Standards, but in exceptional circumstances a technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when a technical committee has collected data of a different kind from that which is normally published as an International Standard (“state of the art”, for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

ISO/IEC TR 12182, which is a Technical Report of type 2, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software engineering*.

Introduction

This Technical Report has several purposes which are directed towards its various intended audiences: the software engineering community; the users of software engineering standards, specifically those developed by ISO/IEC JTC 1/SC 7; and developers of software engineering standards, primarily SC 7.

For the software engineering community, this document will identify the types of software to which particular software engineering standards apply. This will help the software engineer to establish the risk planning criteria, the adequate life-cycle model to apply, the specific effort required for specific life-cycle phases, and the tools needed.

For the users and developers of software engineering standards, it will establish a framework for discussing and identifying candidate software engineering standards based on the software categorization scheme and for using the scheme to relate software and software engineering standards.

Specifically for SC 7, this Technical Report will provide an aid for positioning software engineering standards and work items within the structure of SC 7 and it is intended that new projects, working drafts, committee drafts, and draft international standards will identify the target categorization(s) relevant to the area of application. With respect to the latter, it is understood that, in some documents, only part of this Technical Report will address the specified categorization.

Examples are given to clarify the categorization scheme for software. For the standards developer, this Technical Report helps position existing standards and work items. For the software engineer, it provides a high level scheme with which to locate existing standards.

In addition to the normal introductory clauses, this Technical Report provides a Framework for Categorization of Software, a Scheme of Categorization, and Examples of Application of the Standard.

Information technology — Categorization of software

1 Scope

The scope of this Technical Report is the categories of software (including relevant software development products and data) that are produced by software engineering processes. It describes a categorization scheme for software that encompasses different points of view and significant characteristics and attributes that describe and define software and software categories.

1.1 Field of application

The field of application of the Categorization of Software includes software engineering and its associated standards, software, data, and methodologies.

1.2 Audience and purpose

This Technical Report is primarily directed towards several audiences: the software engineering community; the users of software engineering standards, specifically those developed by ISO/IEC JTC 1/SC 7; and developers of software engineering standards, primarily SC 7.

The purpose or usage of the Categorization of Software is the identifying of software categories, identifying of applicable software engineering standards for software, and determining the relationship of software tasks, processes, or products to software engineering standards.

The use of the Categorization of Software is defined as the generation of a justifiable entry for each of the views specified in Clause 7 for the given software product or in a mapping of a software engineering standard. In some circumstances a null entry is appropriate.

It often occurs that the software engineering processes, and the products of those processes, apply to the procurement or development of certain kinds of software. For example, ISO 6592 applies to large application systems and ISO 9127 applies to packaged software. This Technical Report provides a categorization scheme to assist in (1) the understanding of the area of application of a standard or software, (2) the identification and selection of standards applicable to a software application, and (3) the positioning of new standards.

1.3 Limitations

Since software engineering is a fast changing field, the categorization outlined herein can only be a conceptual scheme. Users, therefore, should use judgement when applying it to applications. The categorization scheme in this Technical Report is empirical in nature. Its formulation is not based on well-defined user needs. The scheme has not been validated in field trials.

2 Conformance

Not applicable.

3 Normative references

No normative references are made within the Categorization of Software.

4 Terms and definitions

This clause provides definitions for terms used in this Technical Report. It does not define the concepts which are explained in the body of the document. It does not define terms which are not specific to the subject matter and are used in accordance with their ordinary English language usage.

For the purposes of this Technical Report, the following terms and definitions apply.

4.1
categorization scheme
an orderly combination of views and categories related to software

4.2
view
a set of related categories

4.3
category
a specifically defined division or grouping of software based upon one or more attributes or characteristics

5 Symbols (and abbreviated terms)

Not applicable.

6 Concept of a categorization of software

The concept of a categorization of software is diagrammatically represented by Figure 1.

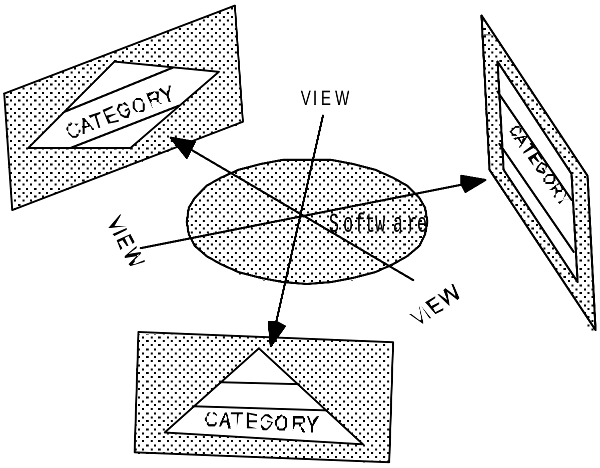

Figure 1 — Categorization of software

As is implied by the figure, the categorization of software is composed of a number of views of software and each view contains categories relevant to the view. The various views are discussed in Clause 7. The selection of categories is at the discretion of the user.

It should be noted that it may be appropriate that a category exists in more than one view and, in several instances, the domain of one category overlaps that of another.

6.1 Structure of views

The categorization scheme consists of 16 views. In use, views of software may be grouped into collections of views such as:

- Internal
  - Operation mode (7.3)
  - Scale of software (7.4)
  - Software stability (7.9)
  - Functionality
    - Software function (7.1)
    - Security requirement (7.13)
  - Reliability requirement (7.14)
  - Required performance (7.12)
  - Primary language (7.6)
- Environment
  - Application area of information system (7.2)
  - Computer system and environment (7.15)
  - User class (7.8)
  - Computer resource requirement (7.16)
  - Software criticality (7.7)
  - Software product availability (7.10)
- Data
  - Data representation (7.5)
  - Usage of software data (7.11)

6.2 Selection of views and categories

For any given instance, such as when the categorization scheme is applied to other software engineering standards, the use of all views may not be required. In these cases, one view or a selected set of views with specific categories will be sufficient to characterize the software. Applications for which subsets of views may be desirable include characterization of software with respect to specific software engineering standards, selection of appropriate software engineering standards, selection of methods, definition of life cycle, determination of document structure, and quality rating.

When applying the characterization scheme to applications for which subsets of views are desirable, all relevant views and related categories should be used. For example, if software is to be characterized with respect to its environment, the set of views contained in the Environment subset presented in 6.1 would be appropriate. Depending upon the particular circumstances of the application, it may be necessary to also include additional views such as security requirement or reliability requirement.

In some applications of the categorization scheme, characterization with respect to a single primary view may be sufficient. For example, software criticality may be the primary view for quality assurance standards.

For other uses of the categorization scheme, several views with specific categories can be combined to represent more specific characteristics of software. For example, software function and user class may be used to determine the documentation structure.

7 Scheme of categorization

Associated with the description of each view of the categorization scheme is a list of the categories relevant to the view. In most cases, the lists are representative or open-ended rather than exhaustive or complete.

The categories relevant to the views are not necessarily mutually exclusive. While in certain applications only one of these categories may apply, in other applications all may apply.

Similarly, the categories relevant to a view are not necessarily at the same level of abstraction.

Where required for clarity, examples are provided for the usage or application of a view. The examples provided are illustrative and provide only a representative set of possible applications.

Users of the Categorization Scheme should apply judgement when selecting the relevant or appropriate categories for a given application or application area.

7.1 Software function

For the software function view, categories should be defined by the type function to which it is targeted.

Examples of categories of software function are:

- Business transaction processing
- Compiler
- Scientific calculation
- Word processing
- Medical systems
- Control systems

7.2 Application area of information system

For the application area view, categories should be defined by the type or class of external system in which they are contained.

For example, software which is an element of process control systems may be categorized as process control facility software and software which is an element of network systems may be categorized as network control software.

Examples of categories of application area are:

- Scientific
- Home appliance
- Equipment
- Process control facility
- Business enterprise
- Networking system

7.3 Operation mode

For the operation mode view, categories should be defined by the specific processing technique or type adopted by the software system.

Examples of categories of operating mode are:

- Batch processing
- Real-time processing
- Time shared processing
- Parallel processing
- Concurrent processing

7.4 Scale of software

For the scale of software view, categories should be defined either by the size or by the complexity of the software.

For example, size could be determined in terms of the number of source lines of code (SLOC) excluding comments and adjusted for the level of the language (i.e., assembly vs. fortran vs. Ada). Complexity may be determined as a function of an appropriate parameter such as data flow complexity. Scale determinations should be normalized for the application area.

Examples of categories of scale of software are:

- Small
- Medium
- Large

It should be understood that the ranges for the above categories should not be interpreted strictly. Instead, the categories should be considered to represent fuzzy or approximate ranges.

7.5 Data representation

For the data view, categories should be defined by the type of data items, types, and structures.

Examples of categories of data are:

- Sequential
- Relational
- Indexed
- Network
- Object
- Entity
- Formatted file

7.6 Primary language

Because the primary language used in software development generally represent or influence significant characteristics of the software, an indication of the type of primary language should be provided.

Examples of categories for primary language are:

- Traditional (COBOL, FORTRAN, etc.)
- Procedural (C or equivalent)
- Functional (Lisp or equivalent)
- Object Oriented (C++ or equivalent)

7.7 Software criticality

For the software criticality view, categories should be defined by the product integrity level rating with the source of the rating specified and the meaning or implication of the rating indicated. Alternatively, categories may be specified by the degree of influence (global, international, etc.) or impact to society (individual, group, business, etc.) if the software system fails. Failure of the software may have safety (human life, property, etc.) or purpose (game, word processing, accounting, etc.) implications.

If integrity level ratings are not utilized, examples of categories for software criticality are:

- National security
- Human life
- Social chaos or panic
- Organizational security
- Private assets
- Privacy

7.8 User class

For the user class view, categories should be defined by the skill level or characteristics of its intended user class. A user is not necessarily a human.

Examples of categories for user class are:

- Novice
- Intermediate
- Expert
- Frequent
- Occasional
- Other software system
- Hardware

7.9 Software stability

Software should be categorized by its intrinsic evolutionary aspect, or stability, in terms of characteristics of the system of which it is a part.

Examples of categories of software stability are:

- Continually changing
- Incremental change
- Unlikely to change

7.10 Software product availability

For the product availability view, categories should be defined by the type(s) of availability of the software.

Examples of categories for availability are:

- Off-the-Shelf
- Custom
- Public
- Proprietary

7.11 Usage of software data

For the data usage view, categories should be defined by the type of usage for which the software data is intended.

Examples of categories for data usage are:

- Single user (individual)
- Multiple sequential users
- Competing mutually exclusive

7.12 Required performance

For the required performance view, categories should be defined by the performance of the software in terms of capacity, throughput, or turnaround where each category is rated according to degree or level.

Examples of categories for required performance are:

- Capacity
  - High
  - Medium
  - Low
- Turnaround / Response Time
  - Fast
  - Moderate
  - Slow
- Throughput
  - Heavy
  - Medium
  - Light

7.13 Security requirement

For the security requirement view, software should be categorized by the level of unauthorized access protection, audit trail, and robustness provided. Additional categories of security requirement may be specified.

Examples of categories for security requirement are indicated by the table of Figure 2.

	Strong	Medium	Weak
Unauthorized Access Protection			
Audit Trail			
Program & Data Protection			

Figure 2 — Categories of Security Requirement

7.14 Reliability requirement

For the reliability requirement view, software should be categorized by the level of required reliability including maturity, fault tolerance, and recoverability.

Examples of categories for reliability requirement are indicated by the table of Figure 3.

	High	Medium	Low
Maturity			
Fault Tolerance			
Recoverability			

Figure 3 — Categories of Reliability Requirement

7.15 Computer system and environment

For the computer system view, software should be identified with a specific target computer system with which it operates.

Examples of categories of computer system are:

- Microprocessor controlled (including workstations, and personal, laptop, and notebook computers)
- Mainframe computer
- Dedicated micro-programming
- Non-von Neumann machine
- Operating System
- Real-time System

7.16 Computer resource requirement

For the computer resource requirement view, software should be identified with respect to computer requirements. Requirements may be stated in terms of the amount required.

Examples of categories of computer resource requirement are:

- Central Processing Unit requirement
- Main memory requirement
- External memory requirement
- Disk requirement
- Local Area Network requirement

8 Application of the scheme

This clause provides examples of applying the scheme for categorization of software to some possible application areas.

8.1 Application to scope of standards

Documentation standards may be developed for specific Scales of Software such as for large application systems (ISO 6592) and for specific kinds of Software Availability such as packaged software (ISO 9127).

8.2 Application to standards

In some cases, specific contents of a standard identify or apply to specific views or categories of software. To illustrate this, the table presented in Figure 4 relates the six characteristics contained in ISO 9126 to the views of the categorization scheme. In the table, the relationships are identified as either primary (P) or secondary (S).

	ISO 9126 Quality Characteristics					
Categorization of S/W Views	Func	Relia	Usab	Effic	Maint	Port
Software Function	P					
Operation Mode	P					
Application Area of IS	P		S			
Scale of Software				P	P	S
Data Representation				P		S
Software Criticality	P	S				
User Class			P			
Required Performance				P		
Software Stability					P	S
Security Requirement	P					
Reliability Requirement		P				
Computer System	S					S
Computer Resource Req.				S		S
Software Product Availability	S				P	S
Usage of Data	P	S				
Primary Language	P			S		P

Figure 4 — Mapping of ISO 9126 to the Categorization Scheme

8.3 Application to software packages

A common application of the categorization scheme is for categorization of specific software packages. The following example provides the categorization for a hypothetical word processing software package. The included parenthetical notations provide rationale information for the categorization of the package and are illustrative of the justifications which might be applicable to an actual system.

8.4 Example Categorization of: Word Processing Package

- Software Function:

  Word Processor

- Operation Mode:

  Interactive Processing (The user of the software enters text or commands which are processed by the software.)

- Application Area of Information System:

  Business/Personal Enterprise (The target environment is a business enterprise but the program is useful for personal or other applications.)

- Scale of Software:

  Small/Medium (smaller, in SLOC, than most similar packages; enhanced capabilities add some complexity)

- Data Representation:

  Object (for command operation) and Formatted File (for command operation, text I/O, and data storage)

- Software Criticality:

  Assets (time) and Individual Convenience

- User Class:

  Intermediate (The target user is a proficient typist with word processing experience but the program is usable by a novice.)

- Required Performance:

  - Capacity - Medium/High (Document size is limited only by the availability of computer resources.)
  - Turnaround / Response Time - Fast (accepts >50cps data entry with fast screen refresh)
  - Throughput - Light (one document at a time)

- Software Stability:

  Controlled change (frequent updates and new releases)

- Security Requirement:

	Strong	Medium	Weak
Unauthorized Access Protection		–	
Audit Trail			–
Program & Data Protection	–		

(password and data encryption available; audit trail not provided; low failure rate)

- Reliability Requirement:

	High	Medium	Low
Maturity	–		
Fault Tolerance	–		
Recoverability	–		

(package has undergone several major revisions and upgrades; allows for novice mistakes; roll-forward keystrokes on aborted edit)

- Computer System:

Microprocessor controlled - personal/laptop/notebook (specific types, models, and classes should be provided)

- Computer Resource Requirement:
    - Main Memory - RAM (minimum and recommended amounts should be provided)
    - External Memory - Mass Storage (minimum and recommended types and amounts should be provided)

- Software Product Availability

  Commercial Off the Shelf; Proprietary

- Usage of the Software Data

  Single user is the normal case but multiple sequential users are possible without degradation.

- Primary Language

  C++

ICS 35.080

Descriptors: data processing, data processing equipment, computers, computer software, categories, description.

Price based on 14 pages