Disclosure to Promote the Right To Information

Whereas the Parliament of India has set out to provide a practical regime of right to information for citizens to secure access to information under the control of public authorities, in order to promote transparency and accountability in the working of every public authority, and whereas the attached publication of the Bureau of Indian Standards is of particular interest to the public, particularly disadvantaged communities and those engaged in the pursuit of education and knowledge, the attached public safety standard is made available to promote the timely dissemination of this information in an accurate manner to the public.

Indian Standard

DEPENDABILITY MANAGEMENT

PART 3 APPLICATION GUIDE
Section 6 Software Aspects of Dependability

ICS 03.100.40; 03.120.01; 35.080
NATIONAL FOREWORD

This Indian Standard (Part 3/Sec 6) which is identical with IEC 60300-3-6 (1997) 'Dependability management — Part 3: Application guide — Section 6: Software aspects of dependability' issued by the International Electrotechnical Commission (IEC) was adopted by the Bureau of Indian Standards on the recommendations of the Reliability of Electronic and Electrical Components and Equipments Sectional Committee and approval of the Electronics and Information Technology Division Council.

The text of the IEC Standard has been approved as suitable for publication as an Indian Standard without deviations. Certain conventions are, however, not identical to those used in Indian Standards. Attention is particularly drawn to the following:

a) Wherever the words 'International Standard' appear referring to this standard, they should be read as 'Indian Standard'.

b) Comma (,) has been used as a decimal marker while in Indian Standards, the current practice is to use a point (.) as the decimal marker.

In this adopted standard, reference appears to certain International Standards for which Indian Standards also exist. The corresponding Indian Standards which are to be substituted in their places are given below along with their degree of equivalence for the editions indicated:

<table>
<thead>
<tr>
<th>International Standard</th>
<th>Corresponding Indian Standard</th>
<th>Degree of Equivalence</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEC 60050 (191) (1990)</td>
<td>IS 1885 (Part 39) : 1999</td>
<td>Technically Equivalent</td>
</tr>
<tr>
<td>Electrotechnical</td>
<td>Electrotechnical vocabulary:</td>
<td></td>
</tr>
<tr>
<td>Vocabulary (IEV)</td>
<td>Part 39</td>
<td></td>
</tr>
<tr>
<td>— Chapter 191:</td>
<td>Reliability of electronic</td>
<td></td>
</tr>
<tr>
<td>Dependability</td>
<td>and electrical items</td>
<td></td>
</tr>
<tr>
<td>and quality of service</td>
<td>(second revision)</td>
<td></td>
</tr>
<tr>
<td>management and</td>
<td>management and quality</td>
<td></td>
</tr>
<tr>
<td>quality assurance —</td>
<td>assurance — Vocabulary</td>
<td></td>
</tr>
<tr>
<td>Vocabulary</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The technical committee responsible for the preparation of this standard has reviewed the provisions of the following International Standards and has decided that they are acceptable for use in conjunction with this standard:

<table>
<thead>
<tr>
<th>International Standard</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEC 60300-1/ISO 9000-4</td>
<td>Dependability management — Part 1:</td>
</tr>
<tr>
<td></td>
<td>Dependability programme management</td>
</tr>
<tr>
<td></td>
<td>Dependability programme elements and tasks</td>
</tr>
</tbody>
</table>

Only the English text in the International Standard has been retained while adopting it as an Indian Standard and as such the page numbers given here are not the same as in IEC publication.
Indian Standard

DEPENDABILITY MANAGEMENT

PART 3 APPLICATION GUIDE

Section 6 Software Aspects of Dependability

1 Scope

This application guide complements IEC 60300-2 and provides guidance for selection and application of dependability elements and tasks with respect to systems or products containing software.

This application guide is intended for use by project managers, contract administrators, product designers, software developers, dependability specialists, quality specialists, support personnel and system maintainers who contribute to the dependability of products or systems.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this section of IEC 60300-3. At the time of publication, the editions indicated were valid. All normative documents are subject to revision, and parties to agreements based on this section of IEC 60300-3 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.


IEC 61160: 1992, Formal design review
Amendment 1 (1994)

ISO 8402: 1994, Quality management and quality assurance – Vocabulary

3 Definitions

For the purpose of this section of IEC 60300-3, the terms and definitions of IEC 60050(191) and ISO 8402 apply.

4 Software aspects

The software aspects of dependability deal with the specific software issues in the establishment and implementation of a dependability programme for a system or product containing software. Emphasis is placed on achieving dependability in the product design and performance objectives in reliability, maintainability and maintenance support.
In the application of a dependability programme to a system or product, it is important to address the dependability issue from a system view point. A product is an entity which may contain hardware or software components or both. A system is an integrated composite entity, which may include the product, supply material, personnel, and related support facilities and services. The system environment defines the operating conditions and interactions of the system components. The availability performance of the system is measured or assessed to validate the achievement of stated dependability objectives in terms of reliability, maintainability, and maintenance support.

Dependability is a collective measure of the performances of a system in its actual state of application or use, with or without the operation of specific software functions which may form part of an integrated system.

It should be noted that software cannot function by itself but forms part of a system to provide specific application. Software is a medium for realization of a system performance objective. Software is characterized in particular by its application function, operating environment, size, language and complexity, installation and upgrade processes. The software aspects of dependability address the software components within a system in the context of dependability performance of the system. They do not address the quality of the software as a stand alone item. Software quality is described in ISO/IEC 9126 [1].

The software aspects of dependability are associated with the integrity of the software component in system operation. Integrity is an inherent design attribute associated with risk containment. Risk is an undesirable exposure or threat associated with the system operation. Risk is characterized by its probability of occurrence and its impact or consequence of the event outcome. The ability of a system and its software component to contain risk is dependent on the system architecture, fault tolerant design, and the degree of rigour in the application of relevant methods to the software. The integrity level is the assigned risk associated with the system operation which is to be contained. The relationship between dependability and integrity is closely linked by the criticality of software application associated with the assigned integrity levels when dealing with the software affecting system performance.

5 Software life cycle phases and processes

The life cycle of software is very much intertwined with the life cycle of its parent system. A typical relationship of the software life cycle phases and the conventional product life cycle phases in accordance with IEC 60300-1/ISO 9000-4 is shown in annex A. An example for the selection of dependability programme elements associated with the software life cycle phases is presented in annex B.

A software life cycle process is a set of planned activities or tasks performed accordingly to achieve a stated goal or objective of a software project. The process involves activities pertaining to a software product from its conception to the termination of its use. Detailed information is contained in ISO/IEC 12207 [2]. Annex C provides informative notes on the software life cycle processes.

Annex D shows the association of software life cycle processes in a typical dependability programme. IEC 60300-2 describes the various elements of a dependability programme in terms of product life cycle phases. The software life cycle processes do not always relate directly to the product life cycle. The process relationships with respect to time are only approximate as considerable variation may occur between different software projects. For example, in some cases software release has to occur before hardware product manufacturing

* Figures in square brackets refer to the bibliography given in annex F.
can start. However, the relevant dependability elements and tasks are incorporated here to illustrate the essence of time-phase applications in dependability programmes. This time-phase representation provides a useful framework for integrating software elements into a hardware system or product.

There are related dependability and quality activities associated with the implementation of a software project or programme. Annex E provides cross-referencing of IEC 60300-2 dependability programme elements and tasks to ISO 9000-3 [3].

6 Application of dependability programmes to products containing software

6.1 General

The guidance given in IEC 60300-2 provides a basis for dependability programmes for products including those containing software. This standard does not duplicate the basic requirements described in IEC 60300-2 but identifies additional task requirements relevant to software aspects of dependability, to complement IEC 60300-2. Guidance is provided on the selection and application of dependability tasks for formulating and implementing specific dependability programmes of products containing software. The following subclauses correspond to the project or product specific dependability programme elements and tasks described in IEC 60300-2 to facilitate cross-referencing.

6.2 Planning and management

6.2.1 Dependability plans

Dependability plans should be established in accordance with 6.1.1 of IEC 60300-2. A dependability plan for a system or product containing software should be an integrated plan that shows the relevant dependability tasks applicable to the system or product, and its constituent hardware and software components. The dependability plan should form part of the overall project management plan.

Planning for software aspects of dependability should take into consideration:

- the system or product requirements pertaining to the software contained in the system or product;
- specific contract requirements affecting the delivery targets and deployment schedule of the software involved;
- the criticality of the software function affecting system or product performance in actual operating environment;
- the feasibility of reusing pre-developed software or off-the-shelf software products;
- timing and resources for the software development if required by the project;
- functional interface of the software involved;
- documentation requirements;
- software test and system integration requirements;
- software maintenance support requirements;
- software update and release requirements.
6.2.2 Project decision management

Project decision management should be conducted in accordance with 6.1.2 of IEC 60300-2.

The management of a software project should be an integral part of the overall project management for the system or product containing the software. Milestone objectives related to software should reflect a coordinated set of deliverables at scheduled targets to facilitate project decision at management reviews.

6.2.3 Traceability management

Traceability management should be in accordance with 6.1.3 of IEC 60300-2.

Traceability of software development and support efforts, functional test data, software release schedules, and field performance data of the product or service deliverables should reflect the requirements stipulated in the overall dependability plan. Relevant records pertaining to software activities should be maintained along with other project activities to facilitate source identification and data correlation for tracking root cause problems.

6.2.4 Configuration management

Configuration management should be in accordance with 6.1.4 of IEC 60300-2.

A configuration management plan should be established and implemented for the software project. This plan is used for identification, control, status accounting, evaluation, change management, release management and delivery of the software involved in the overall project.

Software configuration management should be a functional deliverable item as part of the system or product configuration management plan of the overall project.

6.3 Contract review and liaison

6.3.1 Contract review

Contract review should be in accordance with 6.2.1 of IEC 60300-2.

Specific issues concerning the software involved in a contract should be reviewed in conjunction with the overall project review process. Specific contract requirements pertaining to the software deliverables are reviewed with the customer for acceptance and, where applicable, with the suppliers of subcontract items. Where discrepancies occur, the specific issues should be resolved and the contract amended to reflect the latest status. Contract review records should be maintained.

6.3.2 Management representative

Management representatives should be selected in accordance with 6.2.2 of IEC 60300-2.

There are two aspects related to the function and responsibility of the management representative; one is at the organizational level, the other is at the project level.
The management representative at the organizational level is responsible for the quality system implementation within the organization. This is in conformance with the ISO 9000 quality management standards. Dependability as a specific technical discipline contributes to the organizational infrastructure in terms of value added dependability management principles and practices. The dependability effort within the organization enhances the overall effectiveness of the quality system. Where necessary, the management representative may seek technical assistance from other quality and dependability specialists for resolution of generic or systemic issues related to the quality system at the organization level. A typical position in an organization for the management representative is the quality director, manager, or administrator.

The management representative at the contract or project level deals with the specific issues related to contract delivery of the product or system. The management representative is responsible for interfacing with the customer and suppliers of a specific project by ensuring the quality and dependability of the deliverables. The management representative at the project level should have adequate knowledge concerning the specific project involved. Where necessary, technical experts may be sought to resolve project or contract related problems. A typical position for the management representative in a project is the project manager, principal or lead engineer.

6.4 Dependability requirements

6.4.1 Specification of dependability requirements

Specification of dependability requirements should be in accordance with 6.3.1 of IEC 60300-2.

Specification of dependability requirements for software should consider:
- operating environment of the system or product containing the software and affecting the functional requirements of the software;
- the criticality of the software application in the system or product to achieve the overall availability performance objective;
- permissible downtime duration and outage frequency allocated or contributed by software problems in relation to the overall system downtime;
- software diagnostic and test coverage requirements;
- software qualification and acceptance requirements;
- software test and system integration requirements;
- software documentation requirements;
- software configuration management requirements;
- software release and update requirements;
- software maintenance requirements.

6.4.2 Requirements interpretation

Requirements interpretation should be in accordance with 6.3.2 of IEC 60300-2.
The requirements interpretation should distinguish:

a) those requirements affecting the overall system or product performance;
b) those requirements related to the intended use or application of the software.

The list of dependability requirements provided in 6.4.1 is not exhaustive. There may also be differences in the requirements interpretation when the same basic software is used in a system or product for different applications or operating in separate environments.

6.4.3 Requirements allocation

Requirements allocation should be in accordance with 6.3.3 of IEC 60300-2.

Emphasis should be placed on the allocation of dependability requirements in accordance with the system structure. Mapping of the dependability of requirements onto the system architecture consisting of hardware and software components is a key process. This top down approach provides a means to allocate proper resources relevant to the total system requirements. It facilitates functional design trade-off, permits rationalization of make/buy decisions, and allows planning and implementation of the appropriate level of effort or the degree of engineering rigour required in the development, acquisition, or support of the software for various levels of critical applications.

6.5 Engineering

6.5.1 Reliability engineering

Reliability engineering should be in accordance with 6.4.1 of IEC 60300-2.

Reliability engineering effort applicable to software is associated with the degree of engineering rigour in the application of relevant methods.

Reliability contribution of software components to a system or product containing the software is highly dependent on the development process and the design of the software.

6.5.2 Maintainability engineering

Maintainability engineering should be in accordance with 6.4.2 of IEC 60300-2.

Maintainability engineering effort applicable to software is associated with the degree of engineering rigour in the application of relevant methods.

6.5.3 Maintenance support engineering

Maintenance support engineering should be in accordance with 6.4.3 of IEC 60300-2.

It should be noted that all software maintenance effort is in response to discovered errors in the software design, changes to the software application requirements, or the perfective maintenance of the software (software enhancement) to reduce a shortcoming in the software implementation rather than reacting to a system failure occurrence.

6.5.4 Testability engineering

Testability engineering should be in accordance with 6.4.4 of IEC 60300-2.

The engineering effort to ensure software testability includes:
- design specifications for testability;
- test methods and standards;
- test coverage of the software component;
- test coverage of system requirements;
- software integration and testing for conformance.

Testability is the extent to which an objective and feasible test can be designed into the software to determine whether a requirement is met. Test coverage is the extent to which the test cases are developed to test the system or the software components for conformance to established requirements.

It should be noted that the purpose of testing during software development is to find faults within the software components. The purpose of diagnostic testing during software maintenance is to determine the root cause of an identified system failure or malfunction.

6.5.5 Human factors engineering

Human factors engineering should be in accordance with 6.4.5 of IEC 60300-2.

Human factors have significant impact on system performance. The level of engineering effort relates directly to the planning, design, and execution of the software involved in system operation. Design guidelines and standards should be used to ensure consistency in software design to facilitate testability and integration. Documented test cases and test procedures should be extended to include human factor elements relating to the operation and maintenance of the software to ensure that the overall dependability requirements of the system are met.

Depending on the criticality of system application, the level of human engineering effort required should be consistent with the project application. The potential impact on its immediate environment in case of a system malfunction due to human error should be explored. Human engineering effort applicable to software is associated with the degree of engineering rigour in the application of relevant methods.

6.6 Externally provided products

6.6.1 Subcontracted products

Requirements dealing with subcontracting and subcontracted products should be considered in accordance with 6.5.1 of IEC 60300-2.

The following issues may be applicable to subcontracted software products for integration into the host system:
- subcontracting complete development of the software component or subsystem;
- acquisition or out-sourcing of off-the-shelf software packages;
- subcontracting for modification of existing software.

Where applicable, the dependability requirements of the host system should be reflected in the subcontract statement of work. Subcontract plans should include schedule and milestone deliverables, supplier monitoring, contract reviews, documentation, acceptance criteria and software maintenance support requirements.
6.6.2 Customer-provided products

Customer-provided products should be considered in accordance with 6.5.2 of IEC 60300-2.

Customer-provided software products may be existing software products or a subsystem needed for integration and operation with the host system under contract for delivery. The following requirements should be considered:

- specifications of the customer-provided software product or subsystem;
- interface requirements;
- integration and test requirements;
- a review process in case of host system failures traceable to customer-provided software faults;
- configuration management requirements.

6.7 Analysis, prediction and design review

6.7.1 General

Software analysis methodology in general is based on practical experience and test data with specific software applications and associated operating environments. Software performance models, including those that emulate reliability performance of software systems, are being formulated for reliability prediction and reliability growth assessment purposes. These models represent the mathematical functions relating to specific software performance parameters to provide a quantitative output using the engineering data input. Hence these software performance models are application specific.

There are no generic models or standards for software analysis and evaluation that will meet all application requirements and contract situations. However, there are numerous industry best practices developed for specific software analysis such as:

- software complexity analysis to estimate the fault content in a given set of software modules;
- analysis of code coverage to determine test completeness;
- correlation of software defects classification for rapid root cause analysis and in-process improvement.

The following subclauses present the standard methods identified in IEC 60300-2.

6.7.2 Fault mode and effects analysis

Fault mode and effects analysis (FMEA) [4] should be conducted, where applicable, in accordance with 6.6.1 of IEC 60300-2.

FMEA [4] applicable to software systems should be dealt with at the architectural design level and cascaded down to the functional level as appropriate for critical applications. At the functional level the software component is treated as a black box. The effect of a software fault should be analysed to determine its criticality to the system operation. Where applicable, quantitative assessment should be conducted to determine the failure impact for design trade-off and system performance improvement.
Examples of software fault modes may include:

- a wrong output is provided for a correct input;
- an incorrect input is not recognized;
- the software is corrupted indicating a significant functional error;
- an infinite loop occurs and no output at all is provided by the software function;
- no output is supplied within the required time interval.

6.7.3 Fault tree analysis

Fault tree analysis (FTA) [5] should be conducted, where applicable, in accordance with 6.6.2 of IEC 60300-2.

FTA is applicable to software at the functional level where the dependency of the software component in the overall system performance may be assessed. FTA is a top-down system approach to determine whether the probability of a lower system branch containing the software component is critical.

FTA could also be conducted in conjunction with other analysis techniques as appropriate for determining the reliability performance of the software function for operation of critical systems. The results of FTA and FMEA could be qualitative or quantitative or both. However, it should be noted that most FMEA and FTA performed to date on software are qualitative.

6.7.4 Stress and load analysis

Stress and load analysis should be conducted, where applicable, in accordance with 6.6.3 of IEC 60300-2.

In the context of software application, stress and load analysis is associated with the speed and capacity for the software function to process information throughput. There are no specific methods or standards established for stress and load analysis for software.

6.7.5 Human factors analysis

Human factors analysis should be conducted, where applicable, in accordance with 6.6.4 of IEC 60300-2.

Human factors analysis is a technical discipline which determines the effects of human errors affecting the design, analysis, execution and maintenance of the software product or system. There are no specific standards established for human factors analysis for software. However, in the application of FMEA, FTA, risk analysis and other applicable techniques, the human factors element could be considered as input for the assessment of dependability performance of systems containing software.

6.7.6 Predictions

Predictions should be conducted, where applicable, in accordance with 6.6.5 of IEC 60300-2.

Predictions associated with software products should consider the application environment, operating load and complexity, architecture of system configuration, and the empirical data used to base the reliability performance predictions of the software products.
There are three generic approaches towards prediction methods for software. The first is based on the software development process properties. The second is based on the software product characteristics. The third is based on empirical data gathered from verification processes and actual operation of the software.

Prediction models derived from software development process properties are influenced by the process parameters. The concept is that the management disciplines used for software development could provide reliability projection targets for the software. In this respect, the process parameters are used as benchmarks for reliability improvement.

Prediction models derived from software product characteristics are influenced by the software product parameters such as form, structure and complexity of the software. Reliability prediction based on such models is generally used for off-the-shelf software product evaluation and comparative analysis.

Prediction models derived from software performance data are influenced by the specific application and operating environment of the software. Statistical methods are used for reliability prediction to estimate reliability growth projection based on observed data.

Prediction methods for software are still evolving. Existing methodologies and models are very much product specific and application oriented. There are no standard methods available for generic application of reliability prediction for software. At this point, reliability contribution from software components within a system is mostly estimated, based on observed system performance data containing the software.

6.7.7 Trade-off analysis

Trade-off analysis should be conducted, where applicable, in accordance with 6.6.6 of IEC 60300-2.

By treating software components as functions within a system or product containing software, conventional methods such as FMEA and FTA could be effectively used for design trade-off, make/buy decisions, and comparative analysis for alternate solutions. Trade-off analysis is used for decision making to select an appropriate software approach, an alternate hardware approach, or a combined hardware and software solution in the design architecture to achieve system performance to meet cost-effective project requirements.

6.7.8 Risk analysis

Risk analysis should be conducted, where applicable, in accordance with 6.6.7 of IEC 60300-2.

IEC 60300-3-9 [6] should be used as a reference for conducting risk analysis.

6.7.9 Formal design review

Formal design review should be conducted in accordance with 6.6.8 of IEC 60300-2.

IEC 61160 should be used as guidance to conduct formal design review.
6.8 Verification, validation and test

6.8.1 Verification, validation and test planning

Verification, validation and test planning should be conducted in accordance with 6.7.1 of IEC 60300-2.

Verification is the confirmation by examination and provision of objective evidence that specified requirements have been fulfilled. In design and development of software, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity.

Activities associated with the verification of software should include:

- contract verification;
- process verification;
- requirements verification;
- design verification;
- code verification;
- integration verification;
- documentation verification.

Validation is the confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled. Validation is normally performed on the final product under defined operating conditions.

For software it is important to confirm that the needs and expectations for the software are understood. For high risk applications where significant resources are being expended on the software, it is essential that necessary actions be taken to validate the user expectations of the software or the system containing software. Some examples of such actions are:

- the use of simulations to understand performance expectations;
- the use of prototypes to clarify the user interactions with the software;
- the use of formal models to clarify the functionality of the proposed software.

Activities associated with the validation of software should include:

- preparation of selected test requirements, test specifications and test cases for analysis of test results;
- ensuring test requirements reflect the intended use of the software;
- testing;
- analysis of test results for conformity.

Examples of software analysis and review include techniques such as code inspection, walk through, formalised descriptions, symbolic executions, programme proving, etc. Examples of software verification and validation testing include techniques such as black box, white box, load testing, statistical testing, reliability growth testing, etc.

Test planning includes all applicable verification and validation activities for the specific software under contract. Test planning should be part of the overall project plan.
6.8.2 Life testing

Life testing is a hardware activity and not applicable to software as a stand alone entity. However, life testing is applicable for life evaluation of products containing combined hardware and software where the software is incorporated to serve as a functional component.

6.8.3 Dependability testing

Dependability testing should be conducted in accordance with 6.7.3 of IEC 60300-2.

Testing related to the evaluation, demonstration and acceptance of the availability performance of the system containing software should be conducted in conjunction with other test activities, where applicable to the project. The data collected should provide adequate information for analysis to determine the availability performance of the system for dependability acceptance.

6.8.4 Reliability growth testing

Reliability growth testing should be conducted in accordance with 6.7.4 of IEC 60300-2.


Specific reliability growth models applicable to software consist of the following elements:
- a representation of the failure process by a set of mathematical formulae incorporating certain parameters;
- a method of estimating the parameters by analysis of previous failure data;
- a method of combining the estimated parametric values with the formulae to obtain numerical estimates of reliability measures.

6.8.5 Production testing

Production testing should be conducted in accordance with 6.7.5 of IEC 60300-2.

Production testing is usually applicable to hardware products and to systems containing both hardware and software components as part of a manufacturing process. Production testing is not applicable to testing software as a stand alone item.

6.8.6 Acceptance testing

Acceptance testing should be conducted in accordance with 6.7.6 of IEC 60300-2.

Acceptance testing is associated with software verification and validation. There are three levels of testing of software for acceptance:

a) testing of each software unit or component to ensure conformance to established specifications or standards;
b) testing of integrated software units and components as an aggregate; this is generally known as integration testing;
c) testing of the software installation for final acceptance and commissioning to ensure that the software works in the system configured to operate in actual environment and established conditions as specified in the contract.
6.8.7 Reliability stress screening

Reliability stress screening in accordance with 6.7.7 of IEC 60300-2 is applicable to hardware products. It is not applicable to software.

6.9 Life cycle cost programme

Life cycle cost programme should be in accordance with 6.8 of IEC 60300-2.

Software life cycle cost should be treated as a cost element in the product or system life cycle cost programme.

6.10 Operation and maintenance support planning

6.10.1 Maintenance support planning

Maintenance support planning should be in accordance with 6.9.1 of IEC 60300-2.

Planning for software maintenance support includes estimation of maintenance effort and assignment of tasks and responsibilities for scheduled and unscheduled software release, software update or modification, and perfecutive maintenance for software enhancement. Maintenance support plans should form part of the overall project support plan. Logistics support, including resource allocation, facility and equipment deployment, documentation, and training of personnel, should also be considered as part of the planning effort.

6.10.2 Installation

Installation should be in accordance with 6.9.2 of IEC 60300-2.

Software installation is the execution of the project plan for software delivery and installation in the system on site, final acceptance and commissioning of the software to operate in the system in its actual environment. In this respect, the software code and databases are initialized, the test routines are executed and terminated as specified in the contract specification and test plan. The installation events and results should be documented to facilitate follow-up actions.

6.10.3 Support services

Support services should be in accordance with 6.9.3 of IEC 60300-2.

Support services for software include the continuous support activities for the maintenance and upgrade of the software in an operating system. Support services may be performed by the supplier of the software, or they may be performed by contracting to a third party, or they may be performed by the users of the software with proper instructions. In all cases, training of support personnel is crucial to the success of the support services.

6.10.4 Support engineering

Support engineering should be in accordance with 6.9.4 of IEC 60300-2.

Support engineering is the engineering effort, the knowledge and skills required by the support personnel to fulfil the requirement in performing the support services.
6.10.5 Spares provisioning

Spares provisioning is a hardware activity and not applicable to software.

6.11 Improvements and modifications

6.11.1 Improvement programmes

Improvement programmes should be in accordance with 6.10.1 of IEC 60300-2.

Improvement of software is related to software maintenance. Examples of improvement programmes related to software could be an upgrade of the software features to provide more storage capabilities, or the simplification of administrative or documentation procedures to achieve cost-effective operations.

In all cases, event data should be maintained to provide indications of improvement trends.

Corrective and perfective maintenance for software improvement should be considered in the maintenance process as identified in annex C.

6.11.2 Modification control

Modification control should be in accordance with 6.10.2 of IEC 60300-2.

Software modification control should conform to the established software configuration management process where appropriate administrative and technical procedures are applied. This is to identify, record, and report on the status of the modification to ensure its completeness, consistency and correctness for maintenance of continuous service quality and effectiveness.

Modification control resulting from corrective and perfective maintenance for software should be considered in the configuration management process as identified in annex C.

6.12 Experience feedback

6.12.1 Data acquisition

Data acquisition should be in accordance with 6.11.1 of IEC 60300-2.

Data acquisition should focus on data collection of product or system performance, primarily from field operation and experience feedback from users. Results from test cases and software verification and validation should be included as part of the data. The data collection system should be simple and adequate to provide the essential data necessary for analysis of availability performance. In an ideal situation, the raw data associated with hardware failures, software faults, and procedural errors should be easily segregated for further analysis. Hence, the design of the data acquisition procedure and data collection system should be considered.
6.12.2 Data analysis

Data analysis should be in accordance with 6.11.2 of IEC 60300-2.

Data analysis is essential to provide availability performance trends and to identify anomalies for initiation of corrective or preventive action, as appropriate. Analysis of software data derived from test cases, test results, field performance data or from other relevant sources could provide valuable insights and information such as monitoring reliability growth, maturity indication for software release and systemic problems for root cause analysis. The objective for data analysis should be clearly stated in the project plan. All analysed data should be interpreted and reviewed for management decisions and follow-up actions to effect the continuous improvement process.

7 Tailoring of dependability programmes

Tailoring is a process for matching the requirements to meet a specific project objective.

The criteria set forth in clause 5 of IEC 60300-2 should form the basis for the tailoring process.

The general tailoring process activities include:

- identification of the project environment reflecting the organizational policy and infrastructure;
- analysis of the contract requirements, criticality and impact of the deliverables, capability and resources available for project implementation;
- selection of applicable dependability elements and tasks relevant to the project;
- documenting the rationale in formalising the tailoring decisions as part of the project plan.

The following provides additional information to facilitate tailoring of dependability programmes applicable to systems or products containing software.

Annex B provides general guidance for the principal (first level) selection and implementation of relevant dependability programme elements for specific products containing software. The selection process takes into consideration:

- the product life cycle phases applicable to the project;
- the dependability elements relevant to that part of the product life cycle phases;
- the associated software life cycle processes related to the dependability elements identified;
- the specific software process activities selected for project implementation.

The association of the software life cycle processes with the product life cycle phases is presented in annex D. This annex also cross-references the respective dependability programme elements and tasks applicable to products containing software according to clause 6 of IEC 60300-2, and the dependability management elements and project generic elements according to IEC 60300-1/ISO 9000-4. The respective software life cycle processes are used to identify software specific activities for further tailoring and refinement (second level) in project task implementation.
Annex C provides explanatory notes on the software life cycle processes. Each applicable process contains a set of specific activities for software project implementation. The tailoring process requires further refinement (second level) in the selection and mapping of the appropriate software process activities into the dependability programme when software components are involved. Detailed information on specific software process activities is included in ISO/IEC 12207 [2].

Cost consideration should be given when tailoring a dependability programme to meet specific project objectives. Dependability effort selected for programme implementation should be rationalized to ensure that the selected activities add value.
Typical relationship of product life cycle phases and software life cycle phases

Product life cycle phases

<table>
<thead>
<tr>
<th>Concept and definition</th>
<th>Design and development</th>
<th>Manufacturing and installation</th>
<th>Operation and maintenance</th>
<th>Disposal</th>
</tr>
</thead>
</table>

Software life cycle phases

- Requirements analysis
- System specification
- High-level design
- Detailed design
- Coding and unit test
- Integration test
- System test
- Acceptance test
- Software release
- Software maintenance and enhancement
- Software retirement

NOTE – This chart shows the relationship of the software life cycle phases and the conventional product life cycle phases in a typical time-phase representation applicable to dependability programmes.
## Annex B
(informative)

### Selection of dependability programme elements

The chart below presents a typical association of the software life cycle phases and the applicable dependability programme elements for first level task identification in the tailoring process as described in clause 7.

<table>
<thead>
<tr>
<th>Product life cycle phases</th>
<th>Software life cycle phases</th>
<th>6.1 Planning and management</th>
<th>6.2 Contract review and liaison</th>
<th>6.3 Dependability requirements</th>
<th>6.4 Engineering</th>
<th>6.5 Externally provided products</th>
<th>6.6 Analysis, prediction and design review</th>
<th>6.7 Verification, validation and test</th>
<th>6.8 Life cycle cost programme</th>
<th>6.9 Operation and maintenance support planning</th>
<th>6.10 Improvement and modifications</th>
<th>6.11 Experiences feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept and definition</td>
<td>Requirements analysis</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Design and development</td>
<td>System specification</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>High-level design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Detailed design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Coding and unit test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Integration test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Manufacturing and installation</td>
<td>System test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Acceptance test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Operation and maintenance</td>
<td>Software release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Software maintenance and enhancement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Disposal</td>
<td>Software retirement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

**NOTE** - This chart presents a typical association of the software life cycle phases and the applicable dependability programme elements for first level task identification in the tailoring process as described in clause 7.
Software life cycle processes

C.1 General

Software life cycle processes are described in ISO/IEC 12207 [2]. These processes apply to the acquisition of systems, software products and services. They are applicable to the supply, development, operation and maintenance of software products. A software product is a set of computer programs, procedures, and associated documentation and data. A process is a set of interrelated activities, which transform inputs into outputs, in the context of process implementation and throughput.

The following provides an overview of the software life cycle processes to facilitate their association with dependability programme elements and tasks.

The software life cycle processes are grouped into:

a) primary life cycle processes;

b) supporting life cycle processes;

c) organizational life cycle processes.

C.2 Primary life cycle processes

The primary life cycle processes are those processes that serve the primary parties who initiate or perform in the development, operation or maintenance of software products. These primary parties are the acquirer, the supplier, the developer, the operator, and the maintainer of software products. There are five primary life cycle processes.

a) The acquisition process defines the activities of the acquirer, that is the organization that acquires a system, software product or software service. The acquisition process activities include:

- initiation of the acquisition;
- request for proposal or tender preparation;
- contract preparation and update;
- supplier monitoring;
- acceptance and completion.

b) The supply process defines the activities of the supplier, that is the organization that provides the system, software product or software service to the acquirer. The supply process activities include:

- initiation of the supply process;
- preparation of response;
- contract;
- planning;
- execution and control;
- review and evaluation;
- delivery and completion.
c) **The development process** defines the activities of the developer, that is the organization that defines and develops the software product. The development process activities include:

- development process implementation;
- system requirements analysis;
- system architectural design;
- software requirements analysis;
- software architectural design;
- software detailed design;
- software coding and testing;
- software integration;
- software qualification testing;
- system integration;
- system qualification testing;
- software installation;
- software acceptance support.

d) **The operation process** defines the activities of the operator, that is the organization that provides the service of operating a software system in its live environment for its users. The operation process activities include:

- operation process implementation;
- operational testing;
- system operation;
- user support.

e) **The maintenance process** defines the activities of the maintainer, that is the organization that provides the service of maintaining the software product; it consists in managing modifications to the software product to keep it current and in operational fitness. The maintenance process activities include:

- maintenance process implementation;
- problem and modification analysis;
- modification implementation;
- maintenance review/acceptance;
- migration from old operating environment to new operating environment;
- software retirement.

**C.3 Supporting life cycle processes**

The supporting life cycle processes are those processes that, individually, each supports another process and forms an integral part with the process it supports. A supporting process is employed and executed, as needed, by another process. A supporting process has its specific application and purpose. It contributes to the success and quality of the overall software project. There are eight supporting life cycle processes.

a) **The documentation process** defines the activities for recording the information produced by a life cycle process. The documentation process activities include:
- documentation process implementation;
- design and development of the documents;
- production of the documents;
- maintenance of the documents.

b) The configuration management process defines the configuration management activities on the administration and control of the baseline software items in a system for their release or modification. The configuration management process activities include:
  - configuration management process implementation;
  - configuration identification;
  - configuration control;
  - configuration status accounting;
  - configuration evaluation;
  - release management and delivery.

c) The quality assurance process defines the activities for objectively assuring that the software products and processes are in conformance with their specified requirements and adhere to their established plans. The quality assurance process activities include:
  - quality assurance process implementation;
  - product assurance;
  - process assurance;
  - assurance of quality systems.

d) The verification process defines the activities for verifying the software products of the software project. The verification process activities include:
  - implementation of the verification process;
  - conducting contract verification, process verification, requirements verification, design verification, code verification, integration verification, and documentation verification as appropriate to the software project.

e) The validation process defines the activities for validating the software products of the software project. The validation process activities include:
  - implementation of the validation process;
  - performing validation by means of test cases and analysis of test results.

f) The joint review process defines the activities for evaluating the status and products of an activity. The joint review process activities include:
  - implementation of the joint review process;
  - project management reviews;
  - technical reviews.

g) The audit process defines the activities for determining compliance with the requirements, plans and contract. The audit process activities include:
  - implementation of the audit process;
  - conducting the audit.
h) **The problem resolution process** defines a process for analyzing and resolving the problems discovered during software development, operation, and maintenance. The problem resolution process activities include:
- implementation of the problem resolution process;
- problem resolution.

C.4 Organizational life cycle processes

The organizational life cycle processes are employed by an organization to establish and implement an underlying infrastructure, which is made up of associated life cycle processes, customer focus, management leadership, facilities and personnel, for continuous improvement of the infrastructure and processes. There are four organizational life cycle processes.

a) **The management process** defines the basic activities of the management, including project management, during a life cycle process. The management process activities include:
- initiation and scope definition;
- planning;
- execution and control;
- review and evaluation;
- closure.

b) **The infrastructure process** defines the basic activities for establishing the underlying structure of a life cycle process. The infrastructure process activities include:
- implementation of the infrastructure process;
- establishment of the infrastructure;
- maintenance of the infrastructure.

c) **The improvement process** defines the basic activities that an organization performs for establishing, measuring, controlling and improving its life cycle process. The improvement process activities include:
- improvement process establishment;
- process assessment;
- process improvement.

d) **The training process** defines the activities for providing adequately trained personnel. The training process activities include:
- process implementation;
- training material development;
- training plan implementation.
### Annex D
(informative)

## Association of the software life cycle processes with the product life cycle phases

<table>
<thead>
<tr>
<th>Software life cycle processes (ISO/IEC 12207)</th>
<th>Dependability programme elements and tasks (IEC 60300-2)</th>
<th>Product life cycle phases (IEC 60300-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept and definition</td>
<td>Design and development</td>
</tr>
<tr>
<td>Primary life cycle processes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1 Acquisition process</td>
<td>6.2 Contract review and liaison</td>
<td>xxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.5 Externally provided products</td>
<td>xxxxxxxxxxxx</td>
</tr>
<tr>
<td>5.2 Supply process</td>
<td>6.2 Contract review and liaison</td>
<td>xxxxxxxxxxxx</td>
</tr>
<tr>
<td>5.3 Development process</td>
<td>6.3 Dependability requirements</td>
<td>xxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.4 Engineering</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.6 Analysis, prediction and design review</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.7 Verification, validation and test</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>5.5 Operation process</td>
<td>6.8 Life-cycle cost programme</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.9 Operation and maintenance support planning</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>5.5 Maintenance process</td>
<td>6.9 Operation and maintenance support planning</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.10 Improvements and modifications</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.11 Experiences feedback</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Supporting life cycle processes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.1 Configuration management process</td>
<td>6.11 Experiences feedback</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>6.1.4 Configuration management</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.2 Configuration management process</td>
<td>6.2 Configuration management</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.3 Quality assurance process</td>
<td>Quality systems(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.4 Verification process</td>
<td>6.7 Verification, validation and test</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.5 Validation process</td>
<td>6.7 Verification, validation and test</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.6 Joint process</td>
<td>6.6.8 Formal design review</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.7 Audit process</td>
<td>Quality systems(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>6.8 Problem resolution process</td>
<td>Management review(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Dependability programme review(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>Organizational life cycle process</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7.1 Management process</td>
<td>Dependability policy(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td></td>
<td>Dependability management</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>7.2 Infrastructure process</td>
<td>Organization(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>7.3 Improvement process</td>
<td>Quality systems(2)</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
<tr>
<td>7.4 Training process</td>
<td>Dependability management</td>
<td>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</td>
</tr>
</tbody>
</table>

1) Project generic elements according to IEC 60300-1/ISO 9000-4.
2) Dependability management elements according to IEC 60300-1/ISO 9000-4.

**NOTE** – This chart presents the association of the software life cycle processes and the applicable dependability programme elements for first level task identification in the tailoring process as described in clause 7. The time-phase duration indicated in the column of product life cycle phases (IEC 60300-1) illustrates typical examples in relation to the dependability elements and tasks according to IEC 60300-2.
## Annex E
(informative)

### Cross-references between IEC 60300-2 and ISO 9000-3

<table>
<thead>
<tr>
<th>Subclause in IEC 60300-2</th>
<th>Subclause in ISO 9000-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1.1 Dependability plans</td>
<td>4.2, 5.4, 5.5</td>
</tr>
<tr>
<td>6.1.2 Project decision management</td>
<td>4.1</td>
</tr>
<tr>
<td>6.1.3 Traceability management</td>
<td>4.1</td>
</tr>
<tr>
<td>6.1.4 Configuration management</td>
<td>6.1</td>
</tr>
<tr>
<td>6.2.1 Contract review</td>
<td>5.2</td>
</tr>
<tr>
<td>6.2.2 Management representatives</td>
<td>4.1</td>
</tr>
<tr>
<td>6.3.1 Specification of dependability requirements</td>
<td>5.3</td>
</tr>
<tr>
<td>6.3.2 Requirements interpretation</td>
<td>5.3</td>
</tr>
<tr>
<td>6.3.3 Requirements allocation</td>
<td>5.3, 5.6</td>
</tr>
<tr>
<td>6.4.1 Reliability engineering</td>
<td>5.6</td>
</tr>
<tr>
<td>6.4.2 Maintainability engineering</td>
<td>5.6</td>
</tr>
<tr>
<td>6.4.3 Maintenance support engineering</td>
<td>5.6</td>
</tr>
<tr>
<td>6.4.4 Testability engineering</td>
<td>5.7</td>
</tr>
<tr>
<td>6.4.5 Human factors engineering</td>
<td>5.6, 5.7</td>
</tr>
<tr>
<td>6.5.1 Subcontracted products</td>
<td>6.7, 6.8</td>
</tr>
<tr>
<td>6.5.2 Customer-provided products</td>
<td>6.7, 6.8</td>
</tr>
<tr>
<td>6.6.1 Fault mode and effects analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.2 Fault tree analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.3 Stress and load analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.4 Human factors analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.5 Predictions</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.6 Trade-off analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.7 Risk analysis</td>
<td>6.5, 6.6</td>
</tr>
<tr>
<td>6.6.8 Formal design review</td>
<td>5.5, 5.6, 6.1, 6.2, 6.3</td>
</tr>
<tr>
<td>6.7.1 Verification, validation and test planning</td>
<td>6.4</td>
</tr>
<tr>
<td>6.7.2 Life testing</td>
<td>5.7</td>
</tr>
<tr>
<td>6.7.3 Dependability testing</td>
<td>5.7, 5.8, 6.4</td>
</tr>
<tr>
<td>6.7.4 Reliability growth testing</td>
<td>5.7, 5.8, 6.4</td>
</tr>
<tr>
<td>6.7.5 Production testing</td>
<td>5.7, 5.8, 6.4</td>
</tr>
<tr>
<td>6.7.6 Acceptance testing</td>
<td>5.7, 5.8, 6.4</td>
</tr>
<tr>
<td>6.7.7 Reliability stress screening</td>
<td>5.7, 5.8, 6.4</td>
</tr>
<tr>
<td>6.8 Life cycle cost programme</td>
<td>4.2</td>
</tr>
<tr>
<td>6.9.1 Maintenance support planning</td>
<td>5.10</td>
</tr>
<tr>
<td>6.9.2 Installation</td>
<td>5.9</td>
</tr>
<tr>
<td>6.9.3 Support services</td>
<td>5.9, 5.10, 6.9</td>
</tr>
<tr>
<td>6.9.4 Support engineering</td>
<td>5.9, 5.10</td>
</tr>
<tr>
<td>6.9.5 Spares provisioning</td>
<td>5.9</td>
</tr>
<tr>
<td>6.10.1 Improvement programmes</td>
<td>4.3, 4.4</td>
</tr>
<tr>
<td>6.10.2 Modification control</td>
<td>4.3, 4.4</td>
</tr>
<tr>
<td>6.11.1 Data acquisition</td>
<td>6.2, 6.3</td>
</tr>
<tr>
<td>6.11.2 Data analysis</td>
<td>6.2, 6.3</td>
</tr>
</tbody>
</table>
Annex F
(informative)

Bibliography


Bureau of Indian Standards

BIS is a statutory institution established under the Bureau of Indian Standards Act, 1986 to promote harmonious development of the activities of standardization, marking and quality certification of goods and attending to connected matters in the country.

Copyright

BIS has the copyright of all its publications. No part of these publications may be reproduced in any form without the prior permission in writing of BIS. This does not preclude the free use, in the course of implementing the standard, of necessary details, such as symbols and sizes, type or grade designations. Enquiries relating to copyright be addressed to the Director (Publications), BIS.

Review of Indian Standards

Amendments are issued to standards as the need arises on the basis of comments. Standards are also reviewed periodically; a standard along with amendments is reaffirmed when such review indicates that no changes are needed; if the review indicates that changes are needed, it is taken up for revision. Users of Indian Standards should ascertain that they are in possession of the latest amendments or edition by referring to the latest issue of 'BIS Catalogue' and 'Standards : Monthly Additions'.

This Indian Standard has been developed from Doc : No. LTD 03 (2025).

Amendments Issued Since Publication

<table>
<thead>
<tr>
<th>Amend No.</th>
<th>Date of Issue</th>
<th>Text Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

BUREAU OF INDIAN STANDARDS

Headquarters:
Manak Bhavan, 9 Bahadur Shah Zafar Marg, New Delhi 110 002
Telephones : 2323 0131, 2323 3375, 2323 9402 Website : www.bis.org.in

Regional Offices:
Central : Manak Bhavan, 9 Bahadur Shah Zafar Marg NEW DELHI 110 002 2323 7617
          2323 3841
Eastern : 1/14 C. I. T. Scheme VII M, V. I. P. Road, Kankurgachi
          KOLKATA 700 054 2337 8499, 2337 8561
          2337 8626, 2337 9120
Northern : SCO 335-336, Sector 34-A, CHANDIGARH 160 022 260 3843
          260 9285
Southern : C. I. T. Campus, IV Cross Road, CHENNAI 600 113 2254 1216, 2254 1442
          2254 2519, 2254 2315
Western : Manakalaya, E9 MIDC, Marol, Andheri (East)
          MUMBAI 400 093 2832 9295, 2832 7858
          2832 7891, 2832 7892
Branches: AHMEDABAD. BANGALORE. BHOPAL. BHUBANESHWAR. COIMBATORE.
          FARIDABAD. GHAZIABAD. GUWAHATI. HYDERABAD. JAIPUR. KANPUR.
          LUCKNOW. NAGPUR. NALAGARH. PATNA. PUNE. RAJKOT. THIRUVANANTHAPURAM.
          VISAKHAPATNAM.

Printed at New India Printing Press, Khurja, India