INTERNATIONAL STANDARD

ISO/IEC 15026

First edition
1998-11-15

Information technology — System and software integrity levels

Technologies de l'information — Niveaux d'intégrité du système et du logiciel

Reference number
ISO/IEC 15026:1998(E)

Contents

Page

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. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 15026 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software engineering*.

Information technology — System and software integrity levels

1 Scope

This International Standard introduces the concepts of software integrity levels and software integrity requirements. It defines the concepts associated with integrity levels, defines the processes for determining integrity levels and software integrity requirements, and places requirements on each process. This International Standard does not prescribe a specific set of integrity levels or software integrity requirements. These must be established either on a project by project basis, or for a specific sector and/or country. This International Standard is applicable to software only. The system integrity level and the integrity levels of the non-software components are only required in this International Standard to determine the integrity levels of the software components.

This International Standard is intended for use by developers, users, procurers, and assessors of software products or systems containing software for the administrative and technical support of those products and systems.

A software integrity level denotes a range of values of a software property necessary to maintain system risks within tolerable limits. For software that performs a mitigating function, the property is the reliability with which the software must perform the mitigating function. For software whose failure can lead to a system threat, the property is the limit on the frequency or probability of that failure.

Software integrity requirements are requirements that must be met by the software engineering process used to develop the software, requirements that must be met by the software engineering products, and/or requirements that must be true of the software's performance over time in order to provide a degree of confidence in the software that is commensurate with the software's integrity level.

This International Standard does not prescribe the way in which integrity level determination is integrated with the overall system engineering life cycle processes.

2 Normative references

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

ISO/IEC 2382-1:1993, *Information technology - Vocabulary - Part 1: Fundamental terms.*

ISO/IEC 2382-20:1995, *Information technology - Vocabulary - Part 20: System development.*

ISO 8402:1994, *Quality management and quality assurance - Vocabulary.*

IEC 50-191:1990, *International Electrotechnical Vocabulary, Chapter 191: Dependability and quality of service.*

IEC 300-3-9:1995, *Dependability management - Part 3: Application guide - Section 9: Risk analysis of technological systems.*

ISO/IEC 12207:1995, *Information technology - Software life-cycle processes.*

3 Definitions

For the purposes of this International Standard, the definitions given in ISO/IEC 2382-1, ISO/IEC 2382-20, ISO 8402 and IEC 50-191 apply, except as modified or supplemented by the following definitions.

3.1 component: An entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis.

3.2 degree of confidence: In this standard the term degree of confidence is used only to mean the degree of confidence that software conforms to its requirements.

3.3 design authority: The person or organization that is responsible for producing the design of the system.

3.4 failure: The termination of the ability of an item to perform a required function or its inability to perform within previously specified limits.

3.5 fault isolation: The ability of a subsystem to prevent a fault within the subsystem from causing consequential faults in other subsystems.

3.6 function: An aspect of the intended behavior of the system.

3.7 initiating event: An event that can lead to a threat.

3.8 integrity assurance authority: The independent person or organization responsible for assessment of compliance with the integrity requirements.

3.9 integrity level: A denotation of a range of values of a property of an item necessary to maintain system risks within tolerable limits. For items that perform mitigating functions, the property is the reliability with which the item must perform the mitigating function. For items whose failure can lead to a threat, the property is the limit on the frequency of that failure.

3.10 item: An entity such as a part, component, subsystem, equipment or system that can be individually considered. An item may consist of hardware, software or both.

3.11 mitigating function: A mitigating function is a function that, if provided successfully, will prevent an initiating event from becoming a specified threat.

3.12 risk: A function of the probability of occurrence of a given threat and the potential adverse consequences of that threat's occurrence.

3.13 risk dimension: A perspective from which risk assessment is being made for the system (eg safety, economic, security).

3.14 safety: The expectation that a system does not, under defined conditions, lead to a state in which human life, health, property, or the environment is endangered.

3.15 security: The protection of system items from accidental or malicious access, use, modification, destruction, or disclosure.

3.16 software integrity level: The integrity level of a software item.

3.17 subsystem: A subsystem is any system that is part of a larger system.

3.18 system: An integrated composite that consists of one or more of the processes, hardware, software, facilities and people, that provides a capability to satisfy a stated need or objective.

3.19 systematic failure: A failure related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors.

3.20 system integrity level: The integrity level of a system.

3.21 threat: A state of the system or system environment which can lead to adverse effect in one or more given risk dimensions.

4 Symbols and abbreviations

There are no symbols used in this International Standard. Where appropriate, abbreviations are expressed fully where appearing in the text for the first time.

5 Software integrity levels framework

5.1 How to use this International Standard

The concept of an independent integrity assurance authority is fundamental to the proper use of this International Standard. The integrity assurance authority is the person or organization responsible for certifying compliance with the integrity requirements. Decisions made during negotiation between the design authority and the integrity assurance authority are documented. The decisions to be negotiated include determination of relevant risk dimensions, the specific integrity levels to be used, the specific criteria for assigning each level, the degree of benefit to be allowed for specific architectural features of the design, and the requirements on the software resulting as a consequence of being assigned a particular integrity level.

The processes described in this International Standard are presented as distinct from the overall system engineering processes, but it is not the intention of this International Standard to prevent them from being integrated with the system engineering processes. Regardless of how the processes are implemented, compliance with this International Standard requires that all requirements within it are met.

Subclause 5.2 provides an overview of the processes for determining integrity levels and software integrity requirements. Clauses 6, 7 and 8 describe the processes in more detail and define the requirements imposed on the processes.

5.2 Overview

Figure 1 shows an overview of the processes required to determine system and software integrity levels, and software integrity requirements. Table 1 lists the inputs and outputs for each of the three main processes of system integrity level determination, software integrity level determination, and software integrity requirements determination.

A risk based approach to integrity level determination is used within this International Standard. Therefore the first step in determining the corresponding system integrity level involves the

performance of a risk analysis. IEC Standard 300-3-9 «Risk Analysis of Technological Systems» provides guidance on the performance of risk analysis. Sufficient information must be acquired about the system, its environment and the risk dimensions relevant to the system to allow a risk analysis to be performed. The risks analyzed should cover all relevant risk dimensions such as safety, economic and security as agreed to by the design authority and integrity assurance authority.

Any risks identified by the risk analysis must be evaluated to determine if the risk is tolerable. Once the system design has been analyzed and evaluated to have tolerable risks, a system integrity level is assigned to the system. The system integrity level reflects the worst case risk that is associated with the system.

The integrity level of the software within the system is initially assigned to be the same as the system integrity level. The design of the system may be analyzed to determine if there are architectural features in the system design that would justify assigning a lower integrity level to the software than that of the system.

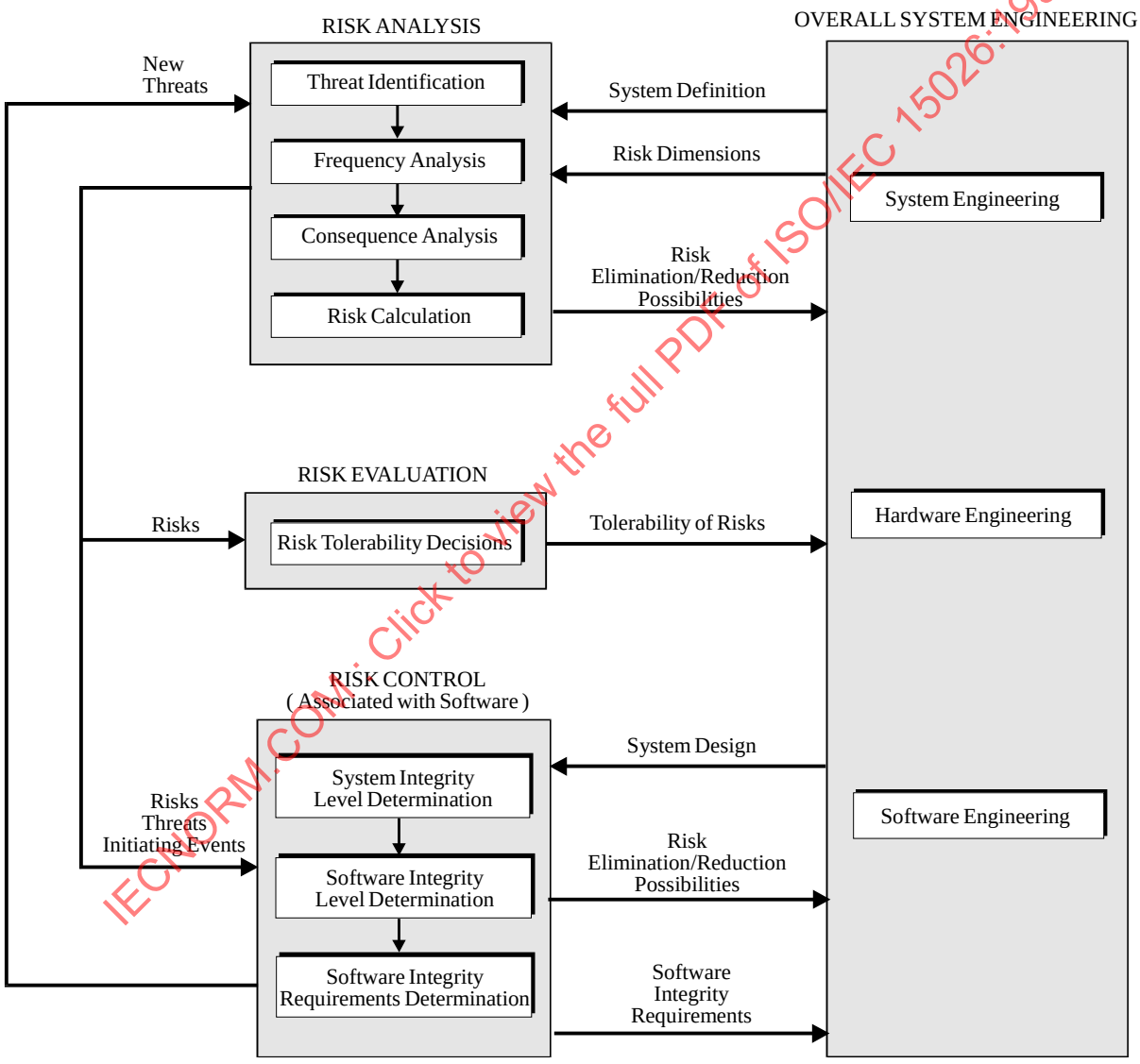

Figure 1 - Overview of Process for Software Integrity Level Determination and Application

Table 1 - Inputs and Outputs

Process	Inputs	Outputs
System Integrity Level Determination	- relevant risk dimensions - system definition - environment definition - system architecture (if available)	- risks - threats - tolerable frequency/probability of threat occurrence - initiating events - initiating event frequency/probability - system integrity level
Software Integrity Level Determination	- system integrity level - subsystem/software architecture - a list of threats and for each threat: - tolerable frequency or probability of occurrence of the threat - initiating events that may lead to the threat - expected frequency or probability of occurrence of each initiating event	- subsystem/software integrity level - architectural features credited with reducing the integrity level
Software Integrity Requirements Determination	- subsystem/software integrity level	- software integrity requirements

A system is an integrated composite that consists of one or more components. A component can be solely software, solely hardware, or can be a subsystem consisting of a further breakdown of components. Initially, any software component of a system is assigned the integrity level of the system. Software integrity level determination involves the analysis of the system architecture to determine if the integrity level for a subsystem can be lowered from that of the system integrity level. This process can be applied recursively until the integrity level for a subsystem consisting solely of software has been determined, or until the integrity level of the subsystem containing software is acceptable to the design authority and integrity assurance authority for assignment to any software component in the subsystem.

It is possible that during the analysis of the architecture that new threats and risks will be identified that were not determined during previous risk analysis. This would require risk analysis to be repeated to take into account the new risk information.

The software integrity level is an assignment which represents either the degree of reliability of provision of a mitigating function, or the required limit on the frequency of failure that could result in a threat. Since software failures are strictly systematic failures, the software integrity level is an indication of the degree of confidence required that either the mitigating function is provided reliably or the failure that would result in a threat will not occur.

The determination and application of software integrity levels is part of the overall process of risk management. Risk management is conducted throughout a system or software product's life, and may be done iteratively as various levels of detail of the design are decided, or as the design evolves. Figure 1 shows the relationships between the overall system engineering processes and the risk management processes of risk analysis, risk evaluation and risk control.

Iteration can occur if the system design is modified to incorporate risk elimination/reduction possibilities, if new threats are identified as the system and software integrity levels are assigned, or if the system design does not result in a tolerable risk.

The software integrity level is used to determine how risk will be controlled, relative to the software components, by identifying what requirements must be true of the software and the software engineering process in order to gain a level of confidence in the software that is commensurate with its role relative to system risks. These requirements are referred to as Software Integrity Requirements.

6 System integrity level determination

The system integrity level corresponds to the tolerable level of risk that is associated with the system. A system can be associated with risk because its failure can lead to a threat, or because its functionality includes mitigation of consequences of initiating events in the system's environment that can lead to a threat. To determine the system integrity level the following steps must be done.

a) Risk analysis determines the level of risk associated with the system. Risk analysis is the process of using available information to identify threats and to estimate the risks associated with the threats.

b) Risk evaluation is the process in which judgments are made on the tolerability of the risks. The outcomes of both risk analysis and risk evaluation can result in changes to the system design to eliminate or reduce risks, which would require the risk analysis and evaluation processes to be repeated.

c) System integrity level assignment is done once a system design that has a tolerable risk has been identified. The exact number of integrity levels from which the selection is made and the criteria for distinguishing between levels shall be negotiated between the Design and Integrity Assurance Authorities.

6.1 Risk analysis

Risk analysis shall be performed to answer three fundamental questions:

What can go wrong? (by threat identification)
How likely is this to happen? (by frequency analysis)
What are the consequences? (by consequence analysis)

With the results of these analyses, the risks can be calculated. The outputs from risk analysis for each identified risk are:

a) an expression of the risk in suitable terms,
b) the threats associated with the risk,
c) the initiating events associated with each threat, and
d) the assumed frequency of initiating event occurrence

The expression of risk resulting from risk analysis is used by the risk evaluation process to determine if the risk is tolerable. All outputs from risk analysis are used by the system and software integrity level determination processes to determine the required integrity level of the software components in the system.

A suitable guide for risk analysis is IEC 300-3-9 ``Risk Analysis of Technological Systems». As safety-specific terminology is used in IEC 300-3-9, the terms "hazard" and «harm» must be interpreted as "threat" and «adverse effect» respectively.

The risk analysis may cover multiple risk dimensions such as safety, economic and security. The specific risk dimensions to be covered shall be determined by the design authority and the integrity assurance authority.

6.1.1 Threat identification

Threats associated with the system should be identified together with the initiating events that could lead to the threats. Threats are associated with a system if the failure of the system could lead to the threat, operation of the system as specified could lead to the threat, or the system performs a mitigating function relative to an initiating event in the environment that could lead to the threat. To identify threats not previously recognized, methods covering the specific situation should be used and documented (see IEC 300-3-9). The system architecture (if available) shall be taken into account during threat identification to ensure that failure modes specific to the technologies used within the system are taken into account.

6.1.2 Frequency analysis

Frequency analysis is used to estimate the likelihood of each initiating event determined by threat identification. When a failure of the system is an initiating event, frequency analysis does not estimate the likelihood of the failure happening, but instead determines a limit on the frequency of failure that is consistent with tolerable risk targets and is practically achievable.

Event frequency shall be estimated using relevant historical data, synthesized using techniques such as fault tree or event tree analysis (see IEC 300-3-9), or estimated using expert judgment.

Frequency analysis done during the system integrity determination process shall treat the system as a black box, and shall not take into account the architecture of the system. The architecture of the system will be taken into account during the software integrity level determination process. The event frequency determined by frequency analysis may be expressed in quantitative terms, or in qualitative terms indicative of ranges of frequencies, such as (Frequent, Probable, Occasional, Remote, Improbable, or Incredible).

Frequency analysis shall take into account that some initiating events will only result in a threat in combination with other events, or as part of a sequence of events.

6.1.3 Consequence analysis

Consequence analysis is used to estimate the severity of the threat should the initiating event(s) occur. Any measures that may mitigate the consequence of the initiating event(s) should be identified. Each threat shall be associated with a consequence category from a set of categories that describe varying degrees of severity of consequence such as (Catastrophic, Major, Severe and Minor).

6.1.4 Risk calculation

The risk associated with each threat is calculated using a risk matrix that relates the frequency of occurrence of initiating events to the severity of consequences of the initiating events. The risk matrix assigns a risk class, such as (High Risk, Intermediate Risk, Low Risk or Trivial Risk) to each combination of frequency and severity. Table 2 shows an example of a risk matrix taken from IEC 300-3-9.

Table 2 - Example Risk Matrix

Frequency of Occurrence	Indicative Frequency (per year)	Severity of Consequence			
		Catastrophic	Major	Severe	Minor
Frequent	>1	High	High	High	Intermediate
Probable	$1 - 10^{-1}$	High	High	Intermediate	Low
Occasional	$10^{-1} - 10^{-2}$	High	High	Low	Low
Remote	$10^{-2} - 10^{-4}$	High	High	Low	Low
Improbable	$10^{-4} - 10^{-6}$	High	Intermediate	Low	Trivial
Incredible	$<10^{-6}$	Intermediate	Intermediate	Trivial	Trivial

The design authority and the integrity assurance authority shall agree upon any risk matrix used to calculate risk. Agreement on the risk matrix requires agreement on the appropriate ranges of initiating event frequencies, agreement on the consequence categories and their definitions, and the risk class associated with each pair of frequency range and consequence category.

6.2 Risk evaluation

The judgment on the tolerability of the risks calculated by the risk analysis process shall be made by the design authority and the integrity assurance authority. In some sectors this judgment is made against regulatory limits established by the sector regulatory agency.

6.3 System integrity level assignment

The system integrity level assigned initially will correspond to the highest risk class assigned to any of the threats associated with the system. The number of integrity levels will correspond to the number of risk classes that were identified. Table 3 shows an example mapping of risk classes to system integrity levels.

Table 3 - Mapping Risk Class to System Integrity Level

Risk Class	System Integrity Level
High	**A**
Intermediate	**B**
Low	**C**
Trivial	**D**

This International Standard does not prescribe the number of, or the denotations used for, the integrity levels.

7 Software integrity level determination

The software integrity level represents an allocation of the system integrity level to a subsystem which contains software as a component or consists solely of software. The software integrity level for a subsystem shall be initially identical to the system integrity level.

This condition remains unless the software integrity level is reduced by taking into account:

a) architectural features of the system that mitigate the consequences of subsystem failures that otherwise would result in a system threat occurrence,
b) architectural features of the system that provide redundancy of mitigating functions so that initiating events are mitigated despite single or multiple failures within subsystems which have been allocated mitigating functions, and
c) whether the subsystem plays any role relative to identified initiating events or mitigating functions.

7.1 Assumptions in software integrity level determination

a) A system integrity level assignment exists for the system.

b) Architectural features of the system shall be defined in sufficient detail to allow identification of subsystem roles and their interfaces.

c) The inputs include:
- the System Integrity Level
- a list of threats and for each threat the following information:
    - the initiating events that may lead to the threat
    - the expected frequency or probability of concurrence of each initiating event
- a definition of the architecture of the system in sufficient detail to determine the role of each subsystem and identify any mitigating features of the architecture.

d) The output is the software integrity level.

7.2 Reduction of software integrity level

To determine a possible reduction of a software integrity level, the following steps shall be performed:

a) identify a set of subsystems which together comprise the complete system.

b) determine if the failure of any subsystem, in isolation or in combination with states of other subsystems, will result in a threat. If the subsystem failure in isolation results in a threat then the integrity level of the subsystem is assigned the same integrity level as the system. If the subsystem failure can only result in a threat in combination with states of other subsystems, then the subsystem integrity level may be reduced depending on the outcome of the assessment defined in subclause 7.3. The assessment in subclause 7.3 is not mandatory, and should only be performed if a lower subsystem integrity level is desired.

c) determine if the failure of any subsystem, in isolation or in combination with states of other subsystems, will result in non-delivery of a mitigating function. If the subsystem failure in isolation results in a non-delivery of a mitigating function then the integrity level of the subsystem is assigned the same integrity level as that assigned to the system. If the subsystem failure can only result in non-delivery of a mitigating function in combination with states of other subsystems, then the subsystem integrity level may be reduced depending on the outcome of the assessment defined in subclause 7.4. The assessment in subclause 7.4 is not mandatory, and should only be performed if a lower subsystem integrity level is desired.

d) determine if there are subsystems with software components, whose failure cannot lead to a threat and whose functionality is not associated with mitigation of any system events. Any such software shall be assigned the lowest integrity level. Fault isolation is necessary in order to ensure that a software failure cannot lead to a threat. The design authority and the integrity assurance authority must agree on the adequacy of architectural features to ensure adequate fault isolation. If the fault isolation is achieved by a failure handling mechanism, the mechanism shall be assigned a software integrity level the same as the system.

The degree of benefit, provided by architectural features in the reduction of software integrity levels from the assigned system integrity level, shall be defined and agreed to by the integrity assurance authority and the design authority.

The process described by steps (a) through (d) is applied recursively until the integrity level for all subsystems consisting solely of software has been determined or until the integrity level of all subsystems containing software is acceptable to the design authority and the integrity assurance authority for assignment to any software component in these subsystems.

7.3 Reducing the software integrity level of software whose failure can result in a threat

For systems whose failure can result in a threat, the system integrity level is partially based upon the limit on the frequency of failure of the system that is consistent with tolerable risk targets. If software failure can only result in a threat in combination with other subsystems being in a particular state, then it may be possible to assign a less stringent limit to the software failure frequency than was used for the system. Frequency analysis may be used to determine the least stringent limit on the software failure frequency that is consistent with tolerable risk targets, and will take into account dependencies between subsystems. The higher software failure frequency limit determined can be used to repeat the risk calculation to determine if there is any change in risk class and thus a lower associated integrity level for the software.

Failure handling mechanisms can be used to detect software failures and to take action to prevent them from becoming initiating events. In these cases the software failure can only become an initiating event if the failure occurs and the failure handling mechanism is ineffective. Examples of failure handling mechanisms are:

- data integrity checks (software mechanism)
- hardware watchdog timers (hardware mechanism)
- manual recovery (human mechanism)

Redundancy can be used to prevent failures from leading to threats, provided common mode failures between the redundant subsystems are avoided. If redundancy is used between software components, then a strategy such as software diversity must be used to avoid common mode failures. When software diversity is used to lower the stringency of the software failure frequency limit, the degree of benefit provided by software diversity and the definition of what constitutes adequate diversity shall be agreed by the design authority and the integrity assurance authority.

7.4 Reducing the software integrity level of software whose failure may result in lack of provision of mitigating functions

If the subsystem failure can only result in non-delivery of a mitigating function in combination with other subsystem states, then the reliability of the software can be lower than that required by the system for the mitigating function. As in subclause 7.3, the techniques used for frequency analysis can be used to determine the degree of reduction of software reliability from the reliability required of the system.