



International
Standard

ISO 19014-4

**Earth-moving machinery —
Functional safety —**

**Part 4:
Design and evaluation of software
and data transmission for safety-
related parts of the control system**

Engins de terrassement — Sécurité fonctionnelle —

*Partie 4: Conception et évaluation du logiciel et de la
transmission des données pour les parties du système de
commande relatives à la sécurité*

**Second edition
2026-10**

Sample Document

get full document from standards.iteh.ai



COPYRIGHT PROTECTED DOCUMENT

© ISO 2026

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword	iv
Introduction	v
1 Scope	1
2 Normative references	1
3 Terms and definitions	1
4 Software development	4
4.1 General	4
4.2 Planning	4
4.3 Artefacts	6
4.4 Software safety requirements specification	7
4.5 Software architecture design	8
4.6 Software module design and coding	8
4.7 Language and tool selection	9
4.8 Software module testing	10
4.9 Software module integration and testing	11
4.10 Safety requirements and protective or risk reduction measures	12
5 Software-based parameterization	13
5.1 General	13
5.2 Data integrity	13
5.3 Software-based parameterization verification	13
6 Transmission protection of safety-related messages on bus systems	13
7 Independence by software partitioning	15
7.1 General	15
7.2 Several partitions within a single microcontroller	16
7.3 Several partitions within the scope of an electronic control unit (ECU) network	18
8 Information for use	18
8.1 General	18
8.2 Instruction handbook	18
Annex A (informative) Description of software methods or measures	20
Annex B (normative) Software validation test environments	32
Annex C (informative) Data integrity assurance calculation	35
Annex D (informative) Methods and measures for transmission protection	37
Annex E (informative) Methods and measures for data protection internal to microcontroller	39
Bibliography	41

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

ISO draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents. ISO shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by ISO/TC 127, *Earth-moving machinery*, Subcommittee SC 2, *Safety, ergonomics and general requirements*, in collaboration with the European Committee for Standardization (CEN) Technical Committee CEN/TC 151, *Construction equipment and building material machines - Safety*, in accordance with the Agreement on technical cooperation between ISO and CEN (Vienna Agreement).

This second edition of ISO 19014-4 cancels and replaces the first edition (ISO 19014-4:2020), which has been technically revised.

The main change is as follows:

— referenced standards have been dated.

A list of all parts in the ISO 19014 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

This document addresses systems comprising any combination of electrical, electronic, and programmable electronic components [electrical, electronic, and programmable electronic systems (E/E/ PES)] used for functional safety in earth-moving machinery.

The structure of safety standards in the field of machinery is as follows.

Type-A standards (basic standards) give basic concepts, principles for design, and general aspects that can be applied to machinery.

Type-B standards (generic safety standards) deal with one or more safety aspects, or one or more types of safeguards that can be used across a wide range of machinery.

- Type-B1 standards on particular safety aspects (e.g. safety distances, surface temperature, noise).
- Type-B2 standards on safeguards (e.g. two-hands controls, interlocking devices, pressure sensitive devices, guards).

Type-C standards (machinery safety standards) deal with detailed safety requirements for a particular machine or group of machines.

This document is a type-C standard as stated in ISO 12100:2010.

This document is of relevance, in particular, for the following stakeholder groups representing market players with regard to machinery safety:

- machine manufacturers (small, medium, and large enterprises);
- health and safety bodies (regulators, accident prevention organisations, market surveillance etc.).

Others can be affected by the level of machinery safety achieved in accordance with this document:

- machine users and employers (small, medium, and large enterprises);
- machine users and employees (e.g. trade unions, organizations for people with special needs);
- service providers, e. g. for maintenance (small, medium, and large enterprises);
- consumers (in case of machinery intended for use by consumers).

The above-mentioned stakeholder groups have been given the possibility to participate at the drafting process of this document.

The machinery concerned and the extent to which hazards or hazardous events are covered are indicated in the Scope of this document.

When the requirements of this type-C standard are different from those in type-A or type-B standards, the requirements of this type-C standard take precedence over the requirements of the other standards for machines that have been designed and built according to the requirements of this type-C standard.

Sample Document

get full document from standards.iteh.ai

Earth-moving machinery — Functional safety —

Part 4:

Design and evaluation of software and data transmission for safety-related parts of the control system

1 Scope

This document specifies general principles for software development and signal transmission requirements of safety-related parts of machine-control systems (MCS) in earth-moving machinery and its equipment, as defined in ISO 6165:2022.

In addition, this document addresses the significant hazards as defined in ISO 12100:2010 related to the software embedded within the MCS. The significant hazards being addressed include incorrect machine control system output responses from machine control system inputs.

To cover all the relevant safety requirements for the products in its scope, this document is used in combination with ISO 19014-1:2026, ISO 19014-2:2026, ISO 19014-3:2026, and ISO 19014-5:2026.

Cybersecurity is out of the scope of this document.

NOTE For guidance on cybersecurity, see an appropriate security standard.

This document is not applicable to earth-moving machinery manufactured before the date of its publication.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 6750-1:2019, *Earth-moving machinery — Operator's manual — Part 1: Contents and format*

ISO 12100:2010, *Safety of machinery — General principles for design — Risk assessment and risk reduction*

ISO 13849-1:2023, *Safety of machinery — Safety-related parts of control systems — Part 1: General principles for design*

ISO 19014-1:2026, *Earth-moving machinery — Functional safety — Part 1: Methodology to determine safety-related parts of the control system and performance requirements*

ISO 19014-2:2026, *Earth-moving machinery — Functional safety — Part 2: Design and evaluation of hardware and architecture requirements for safety-related parts of the control system*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 12100:2010, ISO 13849-1:2023, ISO 19014-1:2026 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at <https://www.iso.org/obp>

— IEC Electropedia: available at <https://www.electropedia.org/>

3.1

bus system

subsystem used in an electronic control system for the transmission of *messages* (3.6)

Note 1 to entry: The bus system consists of the system unit (sources and sinks of information), a transmission path or transmission medium (e.g. electrical lines, fibre-optical lines, radio frequency transmission) and the interface between message source or sink and bus electronics (e.g. protocol application specific integrated circuit, transceivers).

3.2

encapsulated bus system

bus system (3.1) comprising a fixed number or a predetermined maximum number of bus participants connected to each other through a transmission medium with well-defined and fixed performance or characteristics

3.3

failure of peer communication

situation in which the communication peer is not available

3.4

unintended message repetition

situation in which the same *message* (3.6) is unintentionally sent again

3.5

message repetition

situation in which the same *message* (3.6) is intentionally sent again

Note 1 to entry: This technique of resending the same message addresses failures such as *message loss* (3.10).

3.6

message

electronic transmission of data

Note 1 to entry: Transmitted data can include user data, address, or identifier data and data to ensure transmission integrity.

3.7

electronic control unit

ECU

electronic device (electronic programmable controller) used in a control system on earth-moving machinery

[SOURCE: ISO 22448:2010, 3.3, modified — The admitted terms "ECM" and "electronic control module" have been removed.]

3.8

reaction time

time from the detection of a safety-related event until the initiation of a safety reaction

3.9

artefact

work products that are produced and used during a project to capture and convey information

3.10

message loss

unintended deletion of a *message* (3.6) due to a fault of a bus participant

3.11

incorrect sequence

unintended modification of the sequence of *messages* (3.6) due to a fault of a bus participant

Note 1 to entry: *Bus systems* (3.1) can contain elements with stored messages, such as first-in, first-out (FIFOs) etc., that can modify the correct sequence.

3.12

message falsification

unintended modification of *messages* (3.6) due to an error of a bus participant or due to errors on the transmission channel

3.13

message retardation

unintended delay or prevention of the safety function, caused by an overload of the transmission path by normal data exchange or by sending incorrect *messages* (3.6)

3.14

alive counter

accounting component initialized with "0" when the object to be monitored is created or restored

Note 1 to entry: The counter increases from time $t-1$ to time t as long as the object is alive. Finally, the alive counter shows the period of time for which the object has been alive within a network.

3.15

black box testing

testing of an object that does not require knowledge of its internal structure or its concrete implementation

3.16

partition

resource entity allocating a portion of memory, input/output devices, and central processing unit usage to one or more *system tasks* (3.21)

Note 1 to entry: The partitions can be assigned to one or more subsystems within the microcontroller network.

3.17

software partitioning

software fault (3.26) containment method consisting of assigning resources to specific software components with the intention of avoiding the propagation of a software fault to multiple *partitions* (3.16)

3.18

software component

one or more *software modules* (3.19)

[SOURCE: ISO 26262-1:2018, 3.157, modified — The word "units" has been replaced with "modules".]

3.19

software module

independent piece of software that can be independently tested and traced to a specification

Note 1 to entry: The software module is an indivisible software component.

3.20

software partitions

runtime environment with separate system resources assigned

3.21

system task

runtime entities that are executed within the resource budget of *partitions* (3.16) and with different priorities

3.22

independence of software components

exclusion of unintended interactions between software components, as well as freedom from impact on the correct operation of a software component resulting from errors of another software component

3.23

operational history

operating data about a software component or a *software module* (3.19) during its time in service

3.24

maximum cycle time

static time to access a communication bus between nodes at a bus or node level

Note 1 to entry: The application of a time-triggered protocol ensures this cycle time is not exceeded.

3.25

maximum response time

fixed time assigned to a system activity to exchange globally-synchronised *messages* (3.6) on a bus in a time-triggered architecture

3.26

software fault

incorrect step, process, or data definition in software which causes the system to produce unexpected results

3.27

impact analysis

documentation that records the understanding and implications of a proposed change

3.28

configuration management process

task of tracking and controlling changes to the *artefacts* (3.9) in the development process

3.29

constant transmission of messages

situation in which the faulty node continually transmits *messages* (3.6) that compromise the operation of the bus

3.30

blocking access to the data bus

situation in which the faulty node does not adhere to the expected patterns of use and makes excessive demands of service, thereby reducing its availability to other nodes

4 Software development

4.1 General

The main objective of the following requirements is to achieve software reliability by means of readable, understandable, testable, and maintainable software. [Clause 4](#) gives recommendations for the design of software and the subsequent related testing. The avoidance of software faults shall be considered during the entire software development process.

Where an existing software component has been developed to a previous standard and demonstrated through application usage and validation to reduce the risk to as low as reasonably practicable, it is not required to update the software life cycle documentation at the software module level.

Machine control software shall conform to the safety requirements of this clause. In addition, the machine control software shall be designed and developed according to the principles of ISO 12100:2010 for relevant but not significant hazards which are not dealt with by this document.

4.2 Planning

A plan shall be developed to define the relationship between the individual phases of the software development and the related artefacts.

Appropriate methods and measures from [Table 3](#) through [Table 9](#) shall be selected for software development according to the machine performance level required (MPL_r).

NOTE MPL_r is determined by the process outlined in ISO 19014-1:2026 or ISO 19014-5:2026.

The MPL_a of a system is the lower of the MPL_a of the hardware and software.

The MPL_r of the system may be achieved by adding, in parallel, two systems of a lower performance level (according to ISO 19014-2:2026). The software requirements for the two subsystems being added in parallel may be one performance level lower than the machine performance level required (MPL_r) of the overall system. This is only allowable when there are no common cause failures between the two systems.

The suitability of the selected methods or measures to the application shall be justified. Justification shall be made at the beginning of each planned development phase. For a particular application, the appropriate combination of methods or measures shall be stated during development planning. Methods or measures not listed in [Table 3](#) through [Table 9](#) may be used.

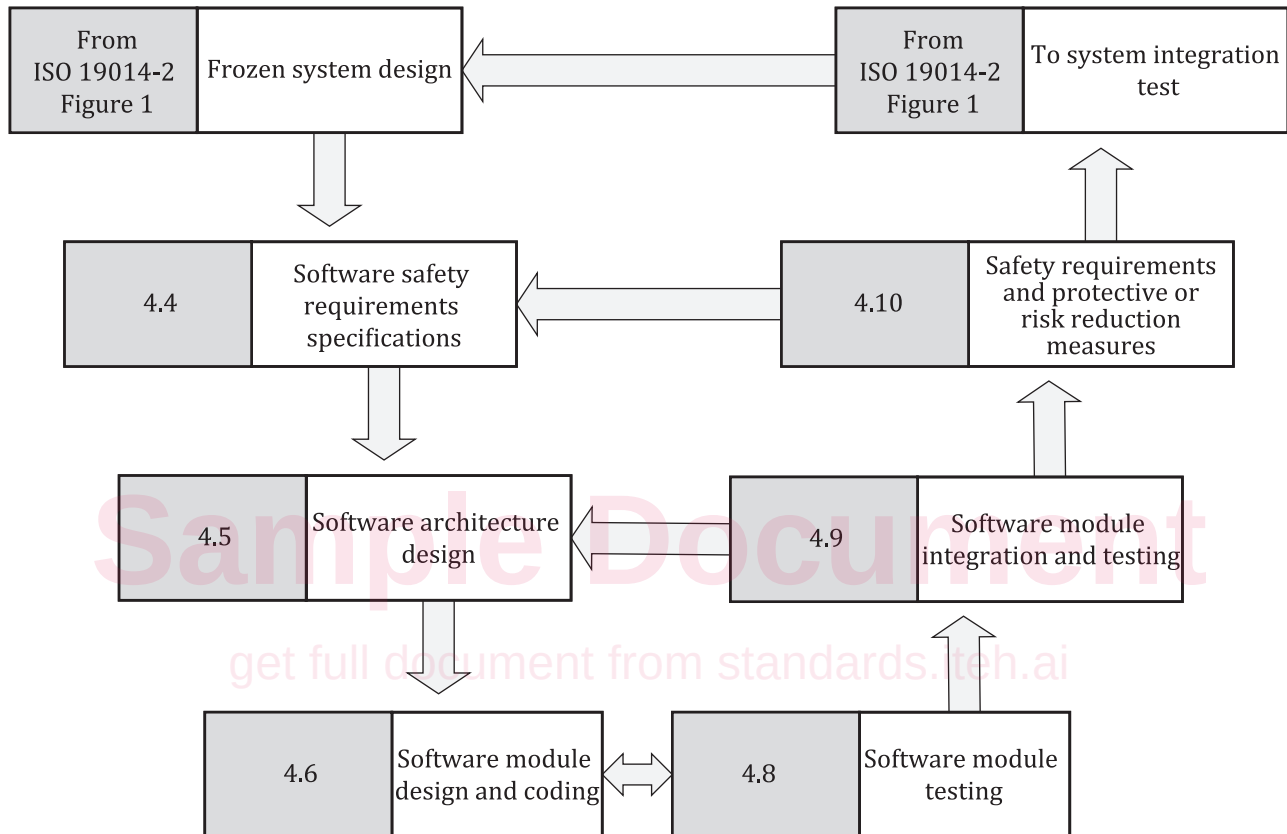


Figure 1 — Software development V-model

[Figure 1](#) is a representation of one possible design method (V-model), which starts with the system design from ISO 19014-2:2026. On the right-hand side of the V-model, the horizontal arrows indicate the requirements against which the software is tested. Any organized, proven development process that meets the requirements of this document may be used for the software development.

When selecting methods and measures, in addition to manual coding, model-based development may be applied where the source code is automatically generated from models.

With each method or measure in the tables, there is a different level of provision for each performance level. [Table 1](#) indicates the requirements.

Table 1 — Software safety requirements specification

Symbol	Software safety requirements specification
+	The method or measure shall be used for this MPL_r . In case this method or measure is not used, the related rationale shall be documented during the safety planning phase.
o	The method or measure may be used for this MPL_r .
–	The method or measure is not suitable to meet this MPL_r .

Methods and measures corresponding to the respective MPL_r shall be selected. Alternative or equivalent methods and measures are identified by letters after the number. At least one of the alternatives or equivalent methods and measures marked with a “+” shall be selected, in which case providing a rationale is not required. An example of this is [Table 2](#).

Table 2 — Example software safety requirements specification

Method or measure		$MPL_r = a$	$MPL_r = b, c$	$MPL_r = d$	$MPL_r = e$
1.a	Measure 1	+	+	–	–
1.b	Measure 2	+	+	+	+
1.c	Measure 3	+	+	+	+

In this case:

- one measure from Measure 1, Measure 2 or Measure 3 shall be fulfilled for $MPL_r = a, b, c$;
- one measure from Measure 2 or Measure 3 shall be fulfilled for $MPL_r = d, e$;
- otherwise, a rationale shall be provided about the unspecified alternative method or measure to satisfy the requirement of the standard for the specific MPL_r .

Rationale or justification shall be provided if other equivalent methods or measures are used instead of the listed methods or measures.

If a software component has any impact on different safety functions with a different MPL_r , then the requirements related to the highest MPL_r shall apply.

If the software contains safety-related and non-safety-related components, then the overall embedded software machine performance level achieved (MPL_a) shall be limited to the software component with the lowest MPL_a . This requirement does not apply when adequate independence between the software components can be demonstrated in accordance with [Clause 7](#).

When reusing a software component that is intended to be modified, an impact analysis shall be performed. An action plan shall be developed and implemented for the overall software life cycle, based on the result of the impact analysis, to ensure that the safety goals are met.

4.3 Artefacts

Once the individual phases of software development plan have been determined, the artefacts shall be defined for each phase to be carried out. Other phases and related artefacts can be added by distributing the activities and tasks. Taking into account the extent and complexity of the project, all artefacts in the individual phases shown in [Figure 1](#) may be modified.

NOTE It is common to combine individual phases if the method or measure used makes it difficult to clearly distinguish between the phases. For example, the design of the software architecture and the software implementation can be generated successively with the same computer-aided development tool, as is done in the model-based development process.

As part of the software development process, the artefacts shall be:

- a) documented according to the outcomes expected from the planned phases;