INTERNATIONAL DOCUMENT

OIML D 31

Edition 2019 (E)

Consolidated Edition including Amendment 1 (2020-09-08)

General requirements for software controlled measuring instruments

Exigences générales pour les instruments de mesure contrôlés par logiciel

ORGANISATION INTERNATIONALE DE MÉTROLOGIE LÉGALE

INTERNATIONAL ORGANIZATION OF LEGAL METROLOGY

Contents

Foreword **4**

1 Introduction **5**

2 Scope and field of application **5**

3 Terms and definitions **6**

3.1 General terminology 6

3.2 Abbreviations 15

4 Instructions for use of this Document in drafting OIML Recommendations **15**

5 Risk assessment **16**

6 Requirements for measuring instruments with respect to the application of software **17**

6.1 General requirements 17

6.2 Requirements specific for configurations 22

7 Type evaluation **36**

7.1 Software documentation to be supplied for type evaluation 36

7.2 Requirements on the evaluation procedure 37

7.3 Verification and evaluation methods 38

7.4 Software evaluation procedure 47

7.5 Equipment Under Test (EUT) 47

8 Verification of a measuring instrument **48**

8.1 General 48

8.2 Verification methods, test items 48

Annex A Bibliography (Informative) **50**

Annex B Example of a software test report (Informative) **52**

Annex C Remarks on measurement terminology (Informative) **59**

Annex D Index **61**

Foreword

The International Organisation of Legal Metrology (OIML) is a worldwide, intergovernmental organisation whose primary aim is to harmonise the regulations and metrological controls applied by the national metrological services, or related organisations, of its Member States. The main categories of OIML publications are:

- **International Recommendations (OIML R)**, which are model regulations that establish the metrological characteristics required of certain measuring instruments and which specify methods and equipment for checking their conformity. OIML Member States shall implement these Recommendations to the greatest possible extent;
- **International Documents (OIML D)**, which are informative in nature and which are intended to harmonise and improve work in the field of legal metrology;
- **International Guides (OIML G)**, which are also informative in nature and which are intended to give guidelines for the application of certain requirements to legal metrology; and
- **International Basic Publications (OIML B)**, which define the operating rules of the various OIML structures and systems.

OIML Draft Recommendations, Documents and Guides are developed by Project Groups linked to Technical Committees or Subcommittees which comprise representatives from the Member States. Certain international and regional institutions also participate on a consultation basis. Cooperative agreements have been established between the OIML and certain institutions, such as ISO and the IEC, with the objective of avoiding contradictory requirements. Consequently, manufacturers and users of measuring instruments, test laboratories, etc. may simultaneously apply OIML publications and those of other institutions.

International Recommendations, Documents, Guides and Basic Publications are published in English (E) and translated into French (F) and are subject to periodic revision.

Additionally, the OIML publishes or participates in the publication of **Vocabularies (OIML V)** and periodically commissions legal metrology experts to write **Expert Reports (OIML E)**. Expert Reports are intended to provide information and advice, and are written solely from the viewpoint of their author, without the involvement of a Technical Committee or Subcommittee, nor that of the CIML. Thus, they do not necessarily represent the views of the OIML.

This publication – reference OIML D 31, edition 2019 (E) – was developed by Project Group 3 in the OIML Technical Subcommittee TC 5/SC 2 *Software*. It was approved for final publication by the International Committee of Legal Metrology at its 54th meeting in 2019 and will be submitted to the International Conference on Legal Metrology in 2020 for formal sanction.

OIML Publications may be downloaded from the OIML web site in the form of PDF files. Additional information on OIML Publications may be obtained from the Organisation's headquarters:

Bureau International de Métrologie Légale
11, rue Turgot - 75009 Paris - France
Telephone: 33 (0)1 48 78 12 82
Fax: 33 (0)1 42 82 17 27
E-mail: biml@oiml.org
Internet: www.oiml.org

General requirements for software-controlled measuring instruments

1 Introduction

The primary aim of this International Document is to provide OIML Technical Committees and Subcommittees with guidance for establishing appropriate requirements for software-related functionalities in measuring instruments covered by OIML Recommendations.

Furthermore, this International Document can provide guidance to OIML Member States in the implementation of OIML Recommendations in their national laws.

2 Scope and field of application

2.1 This International Document specifies the general requirements applicable to legally relevant software-related functionality and security in measuring instruments and gives guidance for verifying the compliance of an instrument with these requirements.

2.2 This Document shall be taken into consideration by the OIML Technical Committees and Subcommittees as a basis for establishing specific software requirements and procedures in OIML Recommendations applicable to particular categories of measuring instruments (hereafter termed "relevant Recommendations").

2.3 The instructions given in this Document apply only to software-controlled measuring instruments or their components.

Note 1: This Document does not cover all the technical requirements specific to software-controlled measuring instruments; these requirements are to be given in the relevant Recommendation, e.g. for weighing instruments, water meters, etc.

Note 2: This Document addresses some aspects concerning data security. In addition, national regulations for this area need to be considered.

3 Terms and definitions

Some of the definitions used in this Document are in conformity with the International Vocabulary of Metrology - Basic and General Concepts and Associated Terms 3rd Edition (OIML V 2-200:2012 [1]), with the International Vocabulary of Terms in Legal Metrology (OIML V 1:2013 [6]), with the OIML International Document *General requirements for measuring instruments – Environmental conditions* (OIML D 11:2013 [2]) and several ISO/IEC International Standards. For the purpose of this Document, the following definitions and abbreviations apply.

3.1 General terminology

3.1.1 audit trail

continuous data file containing a time stamped information record of events, e.g. changes in the values of the parameters of a measuring instrument or software updates, or other activities that are legally relevant and which may influence the metrological characteristics

[OIML V 1:2013, 6.05]

3.1.2 authentication

checking of the declared or alleged identity of a user, process, or measuring instrument

Note: This may be necessary when checking that downloaded software originates from the owner of the certificate.

3.1.3 authenticity

result of the process of authentication (passed or failed)

3.1.4 built-for-purpose device

device constructed for the specific purpose of a metrological task

Note: Undeclared interfaces to the operating system in this kind of device are inaccessible or non-existent.

3.1.5 checking facility

facility that is incorporated in a measuring instrument and which enables significant defect to be detected and acted upon

Note: "Acted upon" refers to any adequate response by the measuring instrument (luminous signal, acoustic signal, prevention of the measurement process, etc.).

adapted from [OIML V 1:2013, 5.07]

3.1.6 communication interface

part of an instrument that enables information to be passed between measuring instruments, components of measuring instruments or other external systems

Note 1: Communication interfaces can be wired, optical, radio, etc. and they are usually designed to use a specific protocol.

Note 2: This definition does not include communication between software parts.

3.1.7 cryptographic certificate

dataset containing the public key belonging to a measuring instrument or a person plus a unique identification of the subject, e.g. serial number of the measuring instrument or name or Personal Identification Number (PIN) of the person, plus a date of expiry

3.1.8 cryptographic means

means such as encryption and decryption with the purpose of hiding information from unauthorised persons (see 3.1.13), or hashes and signatures to ensure integrity and authenticity

3.1.9 data domain

location in memory that each program needs for processing data

Note: Data domains may belong to one *software module* only, or to several.

3.1.10 device-specific parameter

legally relevant parameter with a value that depends on the individual instrument

Note: Device-specific parameters comprise adjustment parameters (e.g. span adjustment or other adjustments or corrections) and configuration parameters (e.g. maximum value, minimum value, units of measurement, etc.).

[OIML V 1:2013, 4.12]

3.1.11 durability

ability of the measuring instrument to maintain its performance characteristics over a period of use

[OIML V 1:2013, 5.15]

3.1.12 electronic measuring instrument

measuring instrument intended to measure an electrical or non-electrical quantity using electronic means and/or equipped with electronic parts

Note: For the purpose of this Document, auxiliary equipment, provided that it is subject to metrological control, is considered to be part of the measuring instrument.

[OIML D 11:2013, 3.1]

3.1.13 electronic signature

software means which is added to software or data with the purpose to verify the origin of software or data, i.e. to prove their authenticity, or to check that the software or data are unchanged, i.e. to prove their integrity

Note 1: For electronic signing, a public key system is used in general, i.e. a pair of keys where only one needs to be kept secret; the other may be public.

Note 2: The secret key is used when software or data are secured. The public key is used when software or data are verified before use.

Note 3: The verifying instance may require a cryptographic certificate of the securing instance (see 3.1.7) to be sure of the authenticity of the public key.

3.1.14 error of indication

indication minus a reference quantity value

Note: This reference value is sometimes referred to as a (conventional) true quantity value. See, however, also OIML V 2-200:2012, 2.12, Note 1).

[OIML V 1:2013, 0.04]

3.1.15 error log

continuous data file containing an information record of failures or significant defects that have an influence on the metrological characteristics of the measuring instrument

3.1.16 event

action in which a modification of a measuring instrument parameter, adjustment factor or update of software module is made

[OIML V 1:2013, 6.06]

3.1.17 event counter

non-resettable counter that increments each time an event occurs

3.1.18 executable code

digital information available in the software or firmware installed on the computing system of the measuring instrument/component (EPROM, hard disk, etc.)

Note: This code is interpreted by the central processing unit (CPU) of the measuring instrument and converted into certain logical, arithmetical, decoding or data transporting operations.

3.1.19 fault

difference between the error of indication and the intrinsic error of a measuring instrument

Note 1: Principally, a fault is the result of an undesired change of data contained in or flowing through an electronic measuring instrument.

Note 2: From the definition it follows that a “fault” is a numerical value which is expressed either in a unit of measurement or as a relative value, for instance as a percentage.

[OIML V 1:2013, 5.12]

3.1.20 hash function

(mathematical) function which maps values from a large (possibly very large) domain into a smaller range

Note: A “good” hash function is such that the results of applying the function to a (large) set of values in the domain will be evenly distributed (and apparently at random) over the range.

[ISO/IEC 9594-8:2014] [3]

3.1.21 integrity (of programs, data, or parameters)

assurance that the programs, data, or parameters have not been subjected to any unauthorised or unintended changes while in use, transfer, storage, repair or maintenance

3.1.22 interface

shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical interconnections, signal exchanges, and other characteristics of the units, as appropriate

[ISO 2382-9:1995] [4]

3.1.23 interruptible cumulative measurement

process of cumulative measurement of the quantity value of a measurand that can be easily and rapidly stopped during normal operation

Note 1: Examples include: a) discontinuous totalising automatic weighing instrument, b) fuel dispenser.

Note 2: See also non-interruptible cumulative measurement (3.1.42).

3.1.24 intrinsic error

error of indication, determined under reference conditions

[OIML V 1:2013, 0.06]

3.1.25 legally relevant

subject to legal control

3.1.26 legally relevant parameter

parameter of a measuring instrument/component, (electronic) device, software or a module subject to legal control

Note: The following types of legally relevant parameters can be distinguished: *type-specific parameters* and *device-specific parameters*.

3.1.27 legally relevant software part

all *software modules* of a measuring instrument/component that are subject to legal control

3.1.28 maximum permissible error (of a measuring instrument)

extreme value of a measurement error, with respect to a known reference quantity value, permitted by specifications or regulations for a given measurement, measuring instrument, or measuring system

adapted from [OIML V 1:2013, 0.05]

3.1.29 measuring instrument

device used for making measurements, alone or in conjunction with one or more supplementary devices

adapted from [OIML V 1:2013, 0.10]

3.1.30 measurement

process of experimentally obtaining one or more quantity values that can reasonably be attributed to a quantity

Note 1: Measurement does not apply to nominal properties.

Note 2: Measurement implies comparison of quantities or counting of entities.

Note 3: Measurement presupposes a description of the quantity commensurate with the intended use of a measurement result, a measurement procedure, and a calibrated measuring system operating according to the specified measurement procedure, including the measurement conditions.

adapted from [OIML V 2-200:2012, 2.1]

3.1.31 measurement data

data used during the measurement process

Note: Measurement data includes measurement result relevant data and measurement process data.

3.1.32 measurement error

measured quantity value minus a reference quantity value

Note 1: The concept of 'measurement error' can be used both

a) when there is a single reference quantity value to refer to, which occurs if a calibration is made by means of a measurement standard with a measured quantity value having a negligible measurement uncertainty or if a conventional quantity value is given, in which case the measurement error is known, and

b) if a measurand is supposed to be represented by a unique true quantity value or a set of true quantity values of negligible range, in which case the measurement error is not known.

Note 2: Measurement implies comparison of quantities or counting of entities.

[OIML V 2-200:2012, 2.16]

3.1.33 measurement metadata

metadata related to the measurement process

Note: Measurement metadata includes measurement result relevant metadata and measurement process metadata.

3.1.34 measurement process data

data used during the measurement process to construct the measurement result

Note: Examples of measurement process data include values of measurement parameters, values of connection settings or values of session parameters.

3.1.35 measurement process information

set of values of qualitative or quantitative variables representing the measurement process

Note: Measurement process information includes measurement process data and measurement process metadata.

3.1.36 measurement process metadata

metadata related to the measurement process

Note: Examples of measurement process metadata include format of the measurement parameters, format of the connection settings or format of the session parameters.

3.1.37 measurement result

set of quantity values being attributed to a measurand together with any other available relevant information

Note 1: The relevant information may consist of e.g. measurement uncertainty, date and time of measurement, number of measurement, identification of sensor and in the case where price calculation is part of the legally relevant software, unit price and price to pay.

Note 2: The measurement result (including the measured quantity value according to V 2:200:2012) is used for the legally relevant purpose, e.g. conclusion of a transaction.

adapted from [V 2-200:2012, 2.9]

3.1.38 measurement result relevant data

data used during the process of constructing the measurement result

Note: Examples of measurement result relevant data include digital number or analogue value originating from a sensor or measuring instrument ID, in cases where it is part of the measurement result.

3.1.39 measurement result relevant metadata

metadata related to the construction of the measurement result

Note: Examples of measurement result relevant metadata include format of the digital number or analogue value originating from a sensor, format of the measured quantity value according to V 2:200:2012 or format of the measuring instrument ID, in cases where it is part of the measurement result.

3.1.40 measurement result relevant information

set of values of qualitative or quantitative variables relevant to the measurement result

Note: Measurement result relevant information includes measurement result relevant data and measurement result relevant metadata.

3.1.41 metadata

data about data or data elements, possibly including their data descriptions, and data about data ownership, access paths, access rights and data volatility

[ISO/IEC 2382:2015 Information technology – Vocabulary]

3.1.42 non-interruptible cumulative measurement

cumulative measuring process with no definite end that cannot be stopped and continued again by a user or an operator without falsifying the result of the measurement

Note 1: Examples include: a) continuous totalising automatic weighing instrument, b) heat meter.

Note 2: See also interruptible cumulative measurement (3.1.23).

3.1.43 protective interface

legally relevant software module that handles all data flow to the legally relevant software part to prevent inadmissible influences

3.1.44 sealing

means intended to protect the measuring instrument against any unauthorised modification, readjustment, removal of parts, software, etc.

Note: This may be achieved by hardware, software or a combination of both.

[OIML V 1:2013, 2.20]

3.1.45 securing

means preventing unauthorised access to hardware or software

[OIML V 1:2013, 2.21]

3.1.46 significant defect

event that has an undesirable impact on the compliance of the measuring instrument or a fault

Note: Examples of significant defect include: a) deletion of the audit trail; b) unauthorised parameter changes; c) unauthorised updates.

3.1.47 software examination

technical operation that consists of determining one or more characteristics of the software according to the specific procedure (e.g. analysis of technical documentation or running the program under controlled conditions)

3.1.48 software identification

sequence of readable characters (e.g. version number, checksum) that represents the software or software module under consideration

Note: It can be checked on an instrument whilst in use.

3.1.49 software interface

program code and dedicated data domain; receiving, filtering, or transmitting data between *software modules*

Note 1: A software interface is not necessarily legally relevant.

Note 2: A software interface is an interface between two or more software modules, used to exchange data and transmit commands.

[OIML V 1:2013, 6.03]

3.1.50 software module

software entity such as a program, subroutine, library, parameter or data set, and other objects including their *data domains* that may be in relationship with other entities

Note: The software of measuring instruments consists of one or more software modules.

3.1.51 software protection

protection of measuring instrument software or data domain by a hardware or software implemented seal

Note: The seal must be removed, damaged or broken to obtain access to change software.

[OIML V 1:2013, 6.04]

3.1.52 software separation

separation of the software in measuring instruments, which can be divided into a *legally relevant part* and a legally non-relevant part

Note: These parts communicate via a software interface.

[OIML V 1:2013, 6.02]

3.1.53 source code

computer program written in a form (programming language) that is legible and editable

Note: Source code is compiled or interpreted into executable code.

3.1.54 storage device

device used for storing measurement data after completion of the measurement process and keeping them available for later legally relevant purposes (e.g. the conclusion of a commercial transaction)

[OIML V 1:2013, 6.07]

3.1.55 time stamp

unique value, e.g. in seconds or a date and time string denoting the date and/or time at which a certain measurement or event occurred

3.1.56 transmission of measurement data

electronic transportation of measurement data via communication lines or other means to a receiver where they are further processed

3.1.57 type (pattern) evaluation

conformity assessment procedure on one or more specimens of an identified type (pattern) of measuring instruments which results in an evaluation report or a certificate

[OIML V 1:2013, 2.04]

3.1.58 type-specific parameter

legally relevant parameter with a value that depends on the type of instrument only

Note: Type-specific parameters are part of the legally relevant software.

[OIML V 1:2013, 4.11]

Example:

Considering a measuring instrument of liquids other than water, the range of cinematic viscosity of a turbine is a type-specific parameter, fixed by the type evaluation of the turbine. All the manufactured turbines of the same type use the same range of viscosity.

3.1.59 universal device

device that is not constructed for a specific purpose, but that can be adapted to a metrological task by software

Note: This kind of device might have undeclared interfaces to the operating system.

3.1.60 user interface

interface that enables information to be interchanged between the operator and the measuring instrument or its hardware components or software modules

Note: Examples are switches, keyboard, mouse, display, monitor, printer, touch-screen, software window on a screen including the software that generates it.

3.1.61 verification

provision of objective evidence that a given item fulfils specified requirements

[adapted from OIML V 2-200:2012, 2.44]

3.1.62 verification of a measuring instrument

conformity assessment procedure (other than type evaluation) which results in the affixing of a verification mark and/or issuing of a verification certificate

Note: See also OIML V 2-200:2012, 2.44.

[OIML V 1:2013, 2.09]

3.2 Abbreviations

EUT	Equipment Under Test
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
IT	Information Technology
MPE	Maximum Permissible Error
OIML	International Organisation of Legal Metrology
PG	Project Group

4 Instructions for use of this Document in drafting OIML Recommendations

4.1 The provisions of this Document apply only to new OIML Recommendations and to OIML Recommendations under revision. OIML Project Groups (Technical Committees, Subcommittees) should use this guidance document to establish software-related requirements in addition to the other technical and metrological requirements of the applicable OIML Recommendation.

4.2 All referred documents are subject to revision, and the users of this Document are encouraged to investigate the possibility of applying the most recent editions of the referred documents.

4.3 It is the objective of this Document to provide the Project Groups responsible for drawing up OIML Recommendations with a set of requirements – partly with different (risk) levels – that are suitable to cover the demands of all kinds of measuring instruments and all areas of application. The Project Group shall determine which risk level is suitable, and how to incorporate the relevant portions of this Document into the OIML Recommendation being drafted. In Clause 5 some aid is given for performing this task.

4.4 PGs should define which influence is considered inadmissible for specific types of instruments.

4.5 PGs shall decide on which measurement data need to comply with requirements. The manufacturer shall document the required metadata where necessary.

5 Risk assessment

5.1 This clause is intended as a guide to determine a set of risk levels to be generally applied for tests carried out on software-controlled measuring instruments. It is not intended as a classification with strict limits leading to special requirements, as in the case of an accuracy classification.

Moreover, this Document does not restrict Project Groups from providing risk assessments that differ from those resulting from the guidelines set forth in this Document. Different risk levels may be used in accordance with special limits prescribed in the relevant Recommendations.

5.2 When selecting risk levels for a particular category of instruments and area of application (trade, direct selling to the public, health, law enforcement, etc.), the following aspects can be taken into account:

a) risk of fraud:
   - the consequence and the social and societal impact of malfunction;
   - the value of the goods to be measured;
   - platform used (built-for-purpose or universal devices);
   - exposure to sources of potential fraud (unattended self-service device).

b) required conformity:
   - the practical possibilities for the industry to comply with the prescribed level.

c) required reliability:
   - environmental conditions;
   - the consequence and the social and societal impact of errors.

d) motivation of the defrauder.

e) the possibility to repeat a measurement or to interrupt it.

PGs should consider risk assessment standards when deciding risk levels, e.g. ISO/IEC 27005 [10].

Throughout the requirements clauses (see 6), various examples of acceptable technical solutions are given illustrating the basic level of protection against fraud, conformity, reliability, and type of measurement (marked with (I)). Where suitable, examples with enhanced counter measures are also presented that consider a raised risk level of the aspects described above (marked with (II)).

The examination level and risk level are linked. A deep analysis of the software shall be performed when a raised risk level is required in order to detect software deficiencies or security weaknesses. On the other hand, mechanical sealing (e.g. sealing of the communication port or the housing) should be considered when choosing the examination level.

6 Requirements for measuring instruments with respect to the application of software

6.1 General requirements

At the time of publishing this Document, the general requirements represent the state of the art in information technology (IT). They are in principle applicable to all kinds of software-controlled measuring instruments and components of measuring instruments. They should be considered in all Recommendations. In contrast to these general requirements, the requirements specific for configurations (6.2) deal with technical features that are not common for some kinds of instruments or in some areas of application.

In the examples, where applicable, both normal and raised risk levels are shown. Notation in this Document is as follows:

(I) Technical solution acceptable in case of normal risk level;

(II) Technical solution acceptable in case of raised risk level (see 5).

6.1.1 Software identification

Software of a measuring instrument/component shall be clearly identified. The identification may consist of more than one part, but at least one part shall be dedicated to the legal purpose.

The identification shall be displayed or printed by the measuring instrument:

- on command; or
- during operation; or
- at start-up for a measuring instrument that can be turned off and on again.

If a measuring instrument/component has neither display nor printer, the identification shall be sent via a communication interface, in order to be displayed/printed on another component.

As an exception, an imprint of the software identification on the instrument/component shall be an acceptable solution if it satisfies all of the following conditions:

a) The user interface does not have any control capability to activate the indication of the software identification on the display, or the display does not technically allow the identification of the software to be shown (analog indicating device or electromechanical counter).

b) The instrument/component does not have an interface to communicate the software identification.

c) After production of the instrument/component a change of the software is not possible, or only possible if the hardware is also changed.

It shall be ensured that the software identification is correctly marked on the instrument/component concerned.

The relevant Recommendation should allow or disallow this exception.

If the software is modified in any way, a new software identification is required.

The software identification and the means of identification (e.g. software version, hash value, checksum, CRC) shall be stated in the certificate. Instructions on how to display or print the software identification shall be in the certificate.

Note 1: Each measuring instrument in use shall conform to the certified type. The software identification enables surveillance personnel and persons affected

by the measurement to determine whether the instrument under consideration is conformable.

Note 2: Unless stated otherwise, the term certificate refers to the OIML type examination certificate.

Example:

(I) The software contains a textual string or a number, unambiguously identifying the installed version. This string is transferred to the display of the instrument when a button is pressed, when the instrument is switched on, or cyclically controlled by a timer.

A version number may have the following structure: A.Y.Z. Considering a flow computer; the letter A will represent the version of the core software that is counting pulses; the letter Y will represent the version of the conversion function (none, at 15 °C, at 20 °C); the letter Z will represent the language of the user interface.

(II) The software calculates a checksum of the executable code and presents the result as the identification instead of or in addition to the string in (I).

6.1.2 Correctness of algorithms and functions

The measuring algorithms and functions of a measuring instrument shall be appropriate and functionally correct for the given application and device type (accuracy of the algorithms, price calculation according to certain rules, rounding algorithms, etc.).

The measurement result and accompanying information required by specific Recommendations or by national legislation shall be displayed or printed correctly.

It shall be possible to examine algorithms and functions either by metrological tests, software tests or software examination (as described in 7.3).

No hidden or undocumented functions or parameters shall exist.

6.1.3 Software protection

6.1.3.1 Prevention of misuse

A measuring instrument shall be constructed in such a way that possibilities for unintentional, accidental, or intentional misuse are minimal. In the framework of this Document, this applies especially to the software. The presentation of the measurement results should be unambiguous for all parties affected.

Note: Software-controlled instruments are often complex in their functionality. The user needs good guidance for correct use and for achieving correct measurement results.

Example:

The user is guided by menus. The legally relevant functions are combined into one branch in this menu. If any measurement data might be lost by an action, the user is warned and requested to perform another action before the function is executed. See also 6.2.3.

6.1.3.2 Evidence of intervention

6.1.3.2.1 Software shall be protected in such a way that evidence of any intervention (e.g. software updates, parameters changes) shall be available. Software shall be secured against unauthorised modification, loading, or changes by swapping the memory device. Mechanical sealing or other technical means may be necessary to secure measuring instruments. Audit trails are considered to be part of the legally relevant software and should be protected as such.

Examples:

(I) A measuring instrument consists of two components, one containing the main metrological functions incorporated in a housing that is sealed. The other component is a universal device with an operating system. Some functions such as the indication are located in the software of this device. To prevent swapping of the software on the universal device the transmission of measurement data between the component and the universal device is encrypted. The key for decryption is hidden in a program that is part of the legally relevant software of the universal device. Only this program knows the key and is able to read, decrypt and use the measurement data. Other programs cannot be used for this purpose as they cannot decrypt the measurement data (see also example in 6.2.2.2.4).

(I)/(II) The housing containing the memory devices is sealed or the memory device is sealed on the printed circuit board.

(II) The write-enable input of the device is inhibited by a switch that can be sealed. The circuit is designed in such a way that the write protection cannot be cancelled by a short-circuit of contacts.

6.1.3.2.2 Only clearly documented functions (see 7.1) may be activated by the user interface, which do not influence the metrological characteristics of the instrument.

Note: The examiner decides whether all of these documented functions are acceptable.

Example:

(I)/(II) All inputs from the user interface are redirected to a software module that filters incoming commands. It only allows the commands to trigger the documented functions and discards all others. This module is part of the legally relevant software.

6.1.3.2.3 Parameters that fix the legally relevant characteristics of the measuring instrument shall be protected against modification. If necessary for the purpose of verification of a measuring instrument, displaying or printing of the current parameter settings shall be possible.

Note 1: Device-specific parameters may stay adjustable or selectable after type evaluation. They should be adjustable/selectable only in a special operational mode of the instrument.

Note 2: Device-specific parameters may be classified as those that should be protected (unalterable) and those that may be accessed (adjustable/selectable parameters) by an authorised person, e.g. the instrument owner or product vendor.

Note 3: Type-specific parameters have identical values for all specimens of a type. They are fixed at type evaluation of the instrument.

Example:

(I)/(II) Device-specific parameters to be protected are stored in a non-volatile memory. The write-enable input of the memory is inhibited by a switch that is sealed.

Refer to examples 6.1.3.2.4 1) to 3) in this clause.

6.1.3.2.4 Software protection shall comprise appropriate sealing by mechanical, electronic and/or cryptographic means, making an intervention impossible or evident.

Note: A cryptographic certificate may be used. The software is signed by a trustworthy institution with an electronic signature. The authenticity of the signed software can be verified by using the public key of the trustworthy institution and decrypting the signature of the certificate.

Examples:

1) (I) Electronic sealing. The legally relevant parameters of an instrument can be input and adjusted by a menu item. The software recognises each change and increments an event counter with each event of this kind. This event counter value can be indicated. The initial value of the event counter is marked durably on the instrument. If the indicated value differs from the registered one, the instrument is in an unverified state (equivalent to a broken seal).

2) (I)/(II) The software of a measuring instrument is constructed such (see example 6.1.3.2.1) that there is no way to modify the legally relevant parameters except via a switch protected menu. This switch is mechanically sealed in the inactive position, making modification of the legally relevant parameters impossible.

   To modify the legally relevant parameters, the switch needs to be activated, inevitably breaking the seal by doing so.

3) (II) The software of a measuring instrument is constructed such that there is no way to access the legally relevant parameters except by authorised persons. If a person wants to access the parameter menu item, that person has to insert their smart card containing a personal identification number (PIN) as part of a cryptographic certificate. The software of the instrument is able to verify the authenticity of the personal identification number (PIN) by the certificate and allows the parameter menu item to be entered. The access and any parameter changes are recorded in an audit trail including the identity of the person (or at least of the smart card used).

6.1.4 Support of hardware features

6.1.4.1 Detection of significant defects

The relevant Recommendation may require detection functions for significant defects. In this case, the manufacturer of the instrument shall be required to design checking facilities into the software or hardware parts or provide means by which the hardware parts can be supported by the software parts of the instrument.

If software is involved in the detection of significant defects, an appropriate reaction is required. For example, the relevant Recommendation may prescribe that the instrument/component is deactivated or an alarm/record in an error log is generated in case a significant defect is detected.

The documentation to be submitted for type evaluation shall contain a list of the significant defects that will be detected by the software and the expected reaction and in case needed for understanding its operation, a description of the detecting algorithm.

Examples:

(I) On each start-up the legally relevant software part calculates a checksum of the program code and legally relevant parameters. The nominal value of these checksums has been calculated in advance and stored in the instrument. If the calculated and stored values do not match, the legally relevant software part stops execution.

In case of a non-interruptible cumulative measurement, the checksum is calculated cyclically and controlled by a software timer. In case a failure is detected, the software displays an error message or switches on a failure indicator and records the time of the significant defect in an error log.

(II) On each start-up, the legally relevant software part calculates a value produced by a cryptographic hash function of the program code and legally relevant parameters. The nominal value of the hash has been calculated in advance and stored in the instrument. If the calculated and stored values do not match, the program stops execution.

In case of a non-interruptible cumulative measurement, the hash value is calculated cyclically and controlled by a software timer. In case a failure is detected, the software displays an error message or switches on a failure indicator and records the time of the significant defect in an error log.

6.1.4.2 Durability protection

It is the manufacturer's choice to realise durability protection facilities addressed in OIML D 11:2013 [2] (5.1.3 (b) and 5.4) in software or hardware, or to allow hardware facilities to be supported by software. The relevant Recommendation may suggest appropriate solutions.

If software is involved in durability protection, an appropriate reaction is required. For example, the relevant Recommendation may prescribe that the instrument/component is deactivated or an alarm/report is generated in case durability is detected as being jeopardised.

Example:

(I)/(II) Some kinds of measuring instruments require an adjustment after a prescribed time interval, in order to guarantee the durability of the measurement. The software gives a warning when the maintenance interval has elapsed and even stops measuring, if it has been exceeded for a certain time interval.

6.1.5 Time stamps

The time stamp shall be in a consistent format, allowing for easy comparison of two different records and tracking progress over time.

The time stamp shall be read from the clock of the instrument. Depending on the kind of instrument or on the field of application, setting the clock may be legally relevant and appropriate protection means shall be taken according to the risk level to be applied (see 6.1.3.2.3).

The internal clock of a stand-alone measuring instrument may have a rather large uncertainty if no means are incorporated to synchronise this clock with the universal time coordinated (UTC). Where the specific field of application requires high accuracy information concerning the exact time of the measurement, it may be necessary to improve the reliability of the internal clock using specific means.

Where relevant, PGs may define requirements and test methods for internal clocks.

Example:

(II) The reliability of the internal quartz-controlled clock device of the measuring instrument is enhanced by redundancy. A timer is incremented by the clock of the microcontroller that is derived from another quartz crystal. When the timer value reaches a preset value, e.g. 1 second, a specific flag of the microcontroller is set and an interrupt routine of the legally relevant software part increments a second counter. At the end of e.g. one day the software reads the quartz-controlled clock device and calculates the difference in the seconds counted by the software. If the difference is within predefined limits, the software counter is reset and the procedure repeats; but if the difference exceeds the limits, the software initiates an appropriate error reaction.

6.2 Requirements specific for configurations

6.2.1 General

The requirements given in this clause are based on typical technical solutions in information technology, although they might not be common in all areas of legal applications. When following these requirements, technical solutions are possible that show the same degree of security and conformity to a type as instruments that are not software-controlled.

The following specific requirements are needed when certain technologies are employed in measuring instruments. They shall be considered in addition to those described in 6.1.

In the examples, where applicable, both normal and raised risk levels are shown. Notation in this Document is as follows:

(I) Technical solution acceptable in case of normal risk level;

(II) Technical solution acceptable in case of raised risk level (see 5).

6.2.2 Specification and separation of legally relevant parts and specification of interfaces

This requirement applies if the measuring instrument/component has interfaces for communicating with other instruments/components, with the user, or with other software parts besides the legally relevant parts within a measuring instrument/component.

Legally relevant parts of a measuring instrument – whether software or hardware parts – shall not be inadmissibly influenced by other parts of the measuring instrument.

Recommendations may specify the software/hardware/data or part of the software/hardware/data that are legally relevant.

6.2.2.1 Separation of components

6.2.2.1.1 Components of a measuring instrument that perform legally relevant functions shall be identified, clearly defined and documented. They form the legally relevant part of the measuring instrument.

Note: The examiner decides whether this part is complete and whether other parts of the measuring instrument may be excluded from further evaluation.

Examples:

1) (I)/(II) An electricity meter is equipped with an optical interface for connecting an electronic device to read out the measurement result. The meter stores all the relevant quantities and keeps the results available to be read out for a sufficient time span. In this system, only the electricity meter is the legally relevant instrument. Other legally non-relevant devices may exist and may be connected to the interface that complies with 6.2.2.1.2. Securing of the data transmission itself (see 6.2.5) is not required.

2) (I)/(II) A measuring instrument consists of the following components:

- a digital sensor that calculates the weight or volume;
- a universal device that calculates the price;
- a printer that prints out the measurement result and the price to pay.

All components are connected by a local area network. In this case the digital sensor, the universal device and the printer are legally relevant components and are optionally connected to a merchandise system that is not legally relevant. The legally relevant components fulfil requirement 6.2.2.1.2 and – because of the transmission via the network – also the requirements contained in 6.2.5.

6.2.2.1.2 It shall be demonstrated that the functions and data of components that are legally relevant cannot be inadmissibly influenced by commands received via the interface from the other, legally non-relevant parts.

This implies that there is an unambiguous assignment of each command to all initiated functions or data changes in the component.

Note: If "legally relevant" components interact with other "legally relevant" components, refer to 6.2.5.

Examples:

1) (I)/(II) The software of the electricity meter (see example (1) of 6.2.2.1.1 above) is able to receive commands for selecting the quantities required. It sends the measurement result (including additional measurement result relevant information – e.g. time stamp, unit) back to the requesting device. The software only accepts commands for the selection of valid allowed quantities and discards any other command, sending back only an error message. There may be securing means for the contents of the dataset, but they are not required, as the transmitted dataset is not subject to legal control.

2) (I)/(II) Inside the housing that is sealed there is a switch that defines the operating mode of the electricity meter: one switch setting indicates the secured mode and the other the free mode (securing means other than a mechanical seal are possible; see examples 6.1.3.2.1 and 6.1.3.2.4). When interpreting received commands, the software checks the position of the switch: in the free mode, the command set that the software accepts is extended compared to the secured mode (e.g. it may be possible to adjust the calibration factor by a command that is discarded in the secured mode).

6.2.2.2 Specification and separation of software parts

6.2.2.2.1 All software modules (programs, subroutines, objects, etc.), that perform legally relevant functions or that process legally relevant measurement data, form the legally relevant software part of a measuring instrument/component. The conformity requirement applies to this part and it shall be made identifiable as described in 6.1.1.

If the separation of the software is not possible or needed, the software is legally relevant as a whole.

Example:

(I) A measuring instrument consists of several digital sensors connected to a personal computer that displays the measurement result. The legally relevant software part on the personal computer is separated from the legally non-relevant part by compiling all procedures realising legally relevant functions (including presentation of results) into a dynamically linkable library. One or several legally non-relevant applications may call functions in this library. These functions receive the measurement data from the digital sensors, calculate the measurement result, and display it in a software window.

6.2.2.2.2 If the legally relevant software part communicates with other software parts, a software interface shall be defined. All communications shall be performed exclusively via this interface. The legally relevant software part and the interface shall be clearly documented. All legally relevant functions and data domains of the software shall be described to enable a type evaluation authority to decide on correct software separation.

The software interface consists of program code and dedicated data domains. Defined coded commands or data are exchanged between the software parts by storing to the dedicated data domain by one software part and reading from it by the other. Writing and reading program code is part of the software interface.

Note: Protection from interruptions (delayed execution or blocking by other processes) is addressed in 6.2.2.2.4.

Example:

(I) In examples 6.2.2.2.1 and 6.2.2.2.3, the legally non-relevant application controls the start of the legally relevant procedures in the library. Omitting a call of these procedures would of course inhibit the legally relevant function of the system. Therefore, the following provisions have been made in the example system: The digital sensors send the measurement data in encrypted form. The key for decryption is hidden in the library. Only the procedures in the library know the key and are able to read, decrypt measurement data, and display measurement results.

6.2.2.2.3 There shall be an unambiguous assignment of each command to all initiated functions or data changes in the legally relevant software part. Functions that are triggered through the software interface shall be declared and documented. Only documented functions shall be activated through the software interface.

Examples:

(I) In the example described in 6.2.2.2.1 the software interface consists of the procedures in the library and their parameters and return values. The interface cannot be circumvented e.g. by pointers to internal data. The number and kind of procedures, parameters, and return values is fixed at compile time.

(II) Legally relevant and legally non-relevant software parts run in separate virtual machines on a universal device. Both machines are configured in such a way that any communication between both software parts can only be done via the defined software interface. The setup of the virtual machines, including the method of communication between both, is part of the legally relevant software. The operating system ensures that the configuration cannot be modified without breaking a seal.

6.2.2.2.4 Where the legally relevant software part has been separated from the non-relevant software part, the legally relevant software part shall have priority using the resources over non-relevant software. The legally relevant process shall not be inadmissibly interrupted by legally non-relevant software. The measurement process (realised by the legally relevant software part) shall not be delayed or blocked by other processes.

Examples:

1) (I) A priority level is assigned to the legally relevant function which is higher than for normal processes and which cannot be decreased by a user/operator of the measuring instrument.

2) (I) The software of an electronic electricity meter reads raw measurement data from an analog-digital converter (ADC). For the correct calculation of the measurement result the delay between the "data ready" event from the ADC to finishing buffering of the measurement data is crucial. The raw values are read by an interrupt routine initiated by the "data ready" signal. The instrument is able to communicate via an interface with other electronic devices in parallel served by another interrupt routine (legally non-relevant communication). The priority of the interrupt routine for processing the raw values is higher than that of the communication routine.

3) (II) Legally relevant and legally non-relevant software parts run in separate virtual machines on a universal device. The configuration of the operating system ensures that the virtual machine on which the legally relevant software part runs always has sufficient system resources available for the legally relevant processes.

Examples from 6.2.2.2.1, 6.2.2.2.2, 6.2.2.2.3 (I) and 6.2.2.2.4 1)/2)(I) are acceptable as a technical solution only for a normal risk level (I). If increased protection against fraud or increased conformity is necessary (see 8), software separation alone is not sufficient and additional means are demanded or the whole software should be considered as under legal control.

6.2.3 Shared indications

A display or printout may be employed to present both information from the legally relevant software part and other information. The contents and layout are specific to the kind of instrument and field of application and shall be defined in the relevant Recommendation. If a display or printout is used both for legally relevant and legally non-relevant outputs, the legally relevant information should always be readable, and clearly distinguishable from other information.

Examples:

(I) In the measuring instrument described in the examples 6.2.2.2.1 to 6.2.2.2.4, the measurement results are displayed in a separate software window. The means described in 6.2.2.2.4 guarantee that only the legally relevant software part can read and display the measurement results. The instrument has an operating system with a multiple windows user interface. The window displaying the legally relevant data is generated and controlled by procedures in the legally relevant dynamically linkable library (see 6.2.2.2). During measurement, these procedures check cyclically that the relevant window is still on top of all the other open windows; if not, the procedures place it on top.

(II) In the measuring instrument described in the examples 6.2.2.2.1 to 6.2.2.2.4 the measurement application runs in kiosk mode. The entire display is controlled by the legally relevant software part. Legally non-relevant data is presented in a special part of the display marked as legally non-relevant.

If increased protection against fraud is necessary (II), a printout as an indication alone may not be suitable and additional precautions in the form of hardware and/or software shall be considered. A component should exist with increased securing means that is able to display the measurement results.

6.2.4 Storage of data

6.2.4.1 General

If measurement data are stored for legal purposes the requirements of 6.2.4.2 to 6.2.4.4 apply.

PGs may decide upon appropriate storage conditions for different applications.

6.2.4.2 Completeness of stored data

The measurement data stored shall be accompanied by all relevant information necessary for future legally relevant use.

Examples:

(I)/(II) A stored dataset of the measurement result includes the following entries:

- measured value including unit;
- time stamp of measurement (see 6.1.5);
- place of measurement or identification of the measuring instrument that was used for the measurement;
- unambiguous identification of the measurement, e.g. consecutive numbers enabling assignment to values printed on an invoice.

6.2.4.3 Protection of stored data

The stored measurement data shall be protected by software means to guarantee the authenticity, integrity and, if necessary, correctness of the information concerning the time of measurement. The software that displays or further processes the measurement data and accompanying data or the measurement result shall check the time of measurement, authenticity, and integrity of the data after having read them from the storage. If an irregularity is detected, the data shall be discarded or marked unusable.

Software modules that prepare data for storing, or that check data after reading are considered part of the legally relevant software.

Note: It is appropriate to require a raised risk level when considering a freely accessible storage.

Raised risk levels might require the application of cryptographic methods. If appropriate, means shall be provided whereby cryptographic keys can only be input or read if a seal is broken. Example (I) applies to local storage and Example (II) applies to freely accessible storage.

Examples:

(I) The program of the storing device calculates a CRC32 of the dataset and appends it to the dataset. It uses a secret initial value for this calculation instead of the value given in the standard. This initial value is employed as a key and stored as a constant in the program code. The reading program has also stored this initial value in its program code. Before using the dataset, the reading program calculates the checksum and compares it with that stored in the dataset. If both values match, the dataset is not falsified. Otherwise, the program assumes falsification and discards the dataset.

(II) The storing program that is part of the legally relevant software generates an electronic signature for the stored dataset. It is appended to the stored dataset. The private and public keys used for signing are generated in a hardware security module which protects the private key against manipulation or reading and exports the public key. The reading program verifies the signature with the public key to check the authenticity and integrity of the dataset. To prove the origin of the dataset the reading program needs to know whether the public key really belongs to the storing program. Therefore, the public key is presented on the display of the measuring instrument and can be registered once, e.g. together with the serial number of the instrument when it is verified in the field.

6.2.4.4 Automatic storing

6.2.4.4.1 When, considering the application, data storage is required, measurement data shall be stored automatically when the measurement is concluded, i.e. when the final measurement result used for the legal purpose has been generated.

The storage device shall have sufficient permanency to ensure that the measurement data are not corrupted under normal storage conditions. There shall be sufficient memory storage for the intended application.

When the data necessary for the calculation of the measurement result are relevant for legal purposes, all measurement result relevant data included in the calculation shall be automatically stored with the final value.

Note 1: In the case of cumulative measurements, it may happen that the same data domain (program variable) is used repeatedly. In that case, storage capacity may not be legally relevant.

Note 2: Stored data does not need to be physically localised in one storage unit, as long as all requirements are met.

Note 3: PGs may define the data which needs to be stored with the final value.

6.2.4.4.2 Stored data may be deleted if either

- the transaction is settled, or
- these data are printed by a printing device subject to legal control.

Note: Other general national regulations (e.g. for tax purposes) may contain strict limitations for the deletion of stored measurement data or results. PGs may define alternative conditions for data deletion.

6.2.5 Transmission via communication lines

If measurement data are transmitted before they are used for legal purposes the following requirements apply:

6.2.5.1 Completeness of transmitted data

The measurement data transmitted shall be accompanied by all relevant information necessary for future legally relevant use.

Examples:

(I)/(II) A transmitted dataset for the measurement result includes the following entries:

- measured value including unit;
- time stamp of measurement (see 6.1.5);
- place of measurement or identification of the measuring instrument that was used for the measurement;
- unambiguous identification of the measurement, e.g. consecutive numbers enabling assignment to values printed on an invoice.

6.2.5.2 Protection of transmitted data

The transmitted data shall be protected by software means to guarantee the authenticity, integrity and, if necessary correctness of the information concerning the time of measurement. The software that displays or further processes the measurement data and accompanying data shall check the time of measurement, authenticity, and integrity of the data received from a transmission channel. If an irregularity is detected, the data shall be discarded or marked unusable.

Software modules that prepare measurement data for sending, or that check measurement data after receiving, are considered part of the legally relevant software.

Note: It is appropriate to require a raised risk level when considering an open network.

Raised risk levels might require application of cryptographic methods. Means shall be provided whereby these keys can only be input or read if a seal is broken.

Examples:

(I) The legally relevant software part of the sending device calculates a CRC32 of the dataset. It is appended to the dataset. It uses a secret initial value for this calculation instead of the value given in the standard. This initial value is employed as a key and stored as a constant in the program code. The program that is part of the legally relevant software of the receiving device has also stored this initial value in its program code. Before using the dataset, the program calculates the checksum and compares it with that stored in the dataset. If both values match, the dataset is not falsified. Otherwise, the program assumes falsification and discards the dataset.

(II) The legally relevant software part of the sending device generates an electronic signature for the transmitted dataset. It is appended to the transmitted dataset. The private and public keys used for signing are generated in a hardware security module which protects the private key against manipulation or reading and exports the public key. A program that is part of the legally relevant software of the receiving device verifies the signature with the public key to check authenticity and integrity of the dataset. To prove the origin of the dataset the receiving program needs to know whether the public key really belongs to the transmitting program. Therefore, the public key is presented on the display of the measuring instrument and

can be registered once, e.g. together with the serial number of the instrument when it is verified in the field.

6.2.5.3 Transmission delay or interruption

The measurement shall not be inadmissibly influenced by a transmission delay or interruption. If network services become unavailable or very slow, no measurement data shall be lost. It may be necessary to stop the measurement process to avoid the loss of measurement data. PGs should decide upon appropriate requirements and mechanisms intended to preserve measurement data where transmission interruptions are possible in the relevant application(s).

Note 1: Consideration should be given to distinguish between static and dynamic measurements.

Note 2: Depending on the area of application, and for cases where measurements are easily repeatable, a loss of transmitted data may be acceptable.

Examples:

(I)/(II) The sending instrument/component waits until the receiver has sent an affirmation of correct receipt of the dataset. The sending instrument/component keeps the dataset in a buffer until this affirmation has been received. The buffer has a capacity for more than one dataset, organised as a FIFO (First-in-first-out) queue.

6.2.6 Compatibility of operating systems and hardware

6.2.6.1 General

If an operating system is part of the measuring instrument, requirements according to 6.2.6.2 to 6.2.6.7 shall be met.

Each of the following operating system requirements shall be met by measures on application level, operating system level or a combination of both. For example, the protective interface may be implemented within the legally relevant application, the operating system, the physical layer, etc.

6.2.6.2 Hardware interfaces

Hardware interfaces not equipped with a protective software interface shall not be able to inadmissibly influence the legally relevant software part (e.g. by preventing usage of the interface by means of a physical seal).

Examples:

(I) The legally relevant application routinely checks all open physical interfaces for incoming traffic. In the case of inadmissible input, it inhibits measurements.

(II) All open interfaces are physically protected or disabled by the operating system.

6.2.6.3 Boot process

6.2.6.3.1 If a secure boot process is needed to ensure protection of the legally relevant software part, the requirements of 6.2.6.3.2 to 6.2.6.3.5 apply.

6.2.6.3.2 In order to ensure integrity and authenticity of the legally relevant software part, a chain of trust shall be established over the individual components of the boot process.

6.2.6.3.3 The processing of the chain of trust can be interrupted, as long as its integrity is preserved.

6.2.6.3.4 The boot configuration shall be protected against modifications.

6.2.6.3.5 Booting via open interfaces shall be prohibited.

Examples:

(I) The boot loader is protected by security means, e.g. a secure password.

(II) A TPM (trusted platform module) verifies the signature of the boot loader, the boot loader then verifies the operating system, which in turn verifies and starts the legally relevant application.

6.2.6.4 System resources

The combination of the legally relevant software part and the operating system shall ensure that there are enough resources for the operation of the legally relevant application.

Examples:

(I) The legally relevant application ensures that it has all the resources it requires.

(II) The smallest number of operating system components required to ensure the measuring operation is selected.

6.2.6.5 Protection during use

6.2.6.5.1 The operation of software that is not legally relevant shall not inadmissibly influence the legally relevant application.

6.2.6.5.2 The combination of the legally relevant software part and the operating system shall ensure that the legally relevant display is distinguishable.

6.2.6.5.3 The access control shall be configured in such way that the intended use cannot be inadmissibly influenced.

6.2.6.5.4 The administration tasks of the legally relevant software part shall be protected.

Examples:

(I) All legally relevant files are write-protected and the access permissions are routinely checked by the legally relevant software part. Modifications of the permissions are logged in an audit trail.

(II) The legally non-relevant software runs in a virtually separated environment.

6.2.6.6 Communication with the legally relevant software part

Communication with the legally relevant software part shall be made via protective interfaces.

Examples:

(I) A legally relevant software module interprets all commands reaching the legally relevant software part and discards the inadmissible ones.

(II) The communication via open software interfaces is protected by means of the operating system.

6.2.6.7 Identification and traceability

6.2.6.7.1 The configuration of the operating system shall be identifiable. The identifier shall be displayed by the measuring instrument:

- on command; or
- during operation.

Examples:

1) On a UNIX-type operating system, the configuration consists of legally relevant:
   - kernel modules
   - list of installed packages
   - libraries
   - accounts and user privileges
   - passwords
   - configuration files
   - file read/write/execute permissions

   All of the above is identified by means of a checksum.

2) On a Windows operating system, the configuration consists of legally relevant:
   - kernel modules
   - list of installed packages
   - libraries
   - accounts and user privileges
   - passwords
   - configuration files
   - file read/write permissions
   - registry keys

   Each of the above is identified by means of a checksum.

6.2.6.7.2 Configuration settings of the operating system shall be protected in such a way that evidence of an intervention is available.

Example:

(I)/(II) All changes to the operating system configuration are logged in an audit trail. Each entry of the audit trail contains a time stamp of the modification as well as the identifier of the new configuration.

6.2.6.8 Suitable environment

The manufacturer shall identify the hardware and software environment that is suitable. Minimum resources and a suitable configuration (e.g. processor, memory, specific communication, version of operating system, etc.) necessary for correct functioning shall be declared by the manufacturer and stated in the certificate.

6.2.6.9 Constraints for operation

Technical means shall be provided in the legally relevant software to prevent operation, if the minimum resources or a suitable configuration are not met. The system shall be operated only in the environment specified by the manufacturer for its correct functioning.

Fixing the hardware, operating system, or system configuration of a universal device or even excluding the usage of an off-the-shelf universal device shall be considered in the following cases:

- if high conformity is required;
- if cryptographic algorithms or keys need to be implemented (see 6.2.4 and 6.2.5).

6.2.7 Conformity of manufactured devices to the certified type

The manufacturer shall produce devices and legally relevant software that conform to the certified type and the documentation submitted.

6.2.8 Maintenance and reconfiguration

6.2.8.1 General

Updating the legally relevant software part of a measuring instrument in the field should be considered as

- a modification of the measuring instrument, when exchanging the software with another certified version, or
- a repair of the measuring instrument, when re-installing the same version.

A measuring instrument which has been modified or repaired while in service may require initial or subsequent verification, dependent on national regulations.

Software which does not realise legally relevant functions of the measuring instrument does not require verification after being updated.

6.2.8.2 Applicability of update requirements

Only versions of the legally relevant software part that conform to the certified type are allowed for use (see 6.2.7). They shall be stated in the certificate. Applicability of the following requirements depends on the kind of instrument and is to be worked out in the relevant Recommendation. The following options 6.2.8.3 and 6.2.8.4 are alternatives. In the case that device-specific parameters (especially calibration parameters) are concerned, only a verified update should be done.

This issue concerns verification of a measuring instrument in the field. Refer to clause 8 for additional constraints.

6.2.8.3 Verified update

The software to be updated can be loaded locally, i.e. directly on the measuring instrument, or remotely via a network. Loading and installation may be two different steps (as shown in Fig. 1) or combined into one, depending on the needs of the technical solution. A seal needs to be broken for the update to take effect. A person should be on the installation site of the measuring instrument to check that the updated software has been installed successfully. After the update of the legally relevant software part of a measuring instrument (exchange with another certified version or re-installation) the measuring instrument should not be employed for legal purposes before a verification of the measuring instrument as described in clause 8 has been performed and the securing means have been renewed and the protection means have been reactivated (if not otherwise stated in the relevant Recommendation or in the certificate).

6.2.8.4 Traced update

6.2.8.4.1 The software shall be implemented in the instrument according to the requirements for Traced update (6.2.8.4.2 to 6.2.8.4.8), if it is in compliance with the relevant Recommendation. Traced update is the procedure of changing software in a verified instrument or component after which a subsequent verification is not necessary. This means the traced update shall not affect existing parameters. The software to be updated can be loaded locally, i.e. directly on the measuring instrument, or remotely via a network. The software update is recorded in an audit trail (see 3.1.1). The procedure of a traced update comprises several steps: loading, integrity checking, checking of the origin (authentication), installation, logging and activation.

6.2.8.4.2 Traced update of software shall be automatic. If some of the securing or protection measures of the instrument are turned off to enable updating, they shall be turned on again immediately after update, independent of the result of the update process.

Note: Triggering of the traced update process may require intervention/manual actions by the user of the measuring instrument.

6.2.8.4.3 Software shall be protected in such a way that evidence of any intervention shall be available. During an update, any existing audit trail information and event counter value shall be retained.

Example:

(I) At start-up of the measuring instrument, a checksum over the legally relevant software part is calculated and compared with a nominal value. The instrument only starts if the values match. Otherwise an event counter is increased by 1. During an update, the nominal value is modified to match the new software part. The event counter value is retained and treated by the new software in the same manner as before.

6.2.8.4.4 Technical means shall be employed to guarantee the authenticity of the loaded software, i.e. that it originates from the owner of the certificate.

Example:

(II) The authenticity check is accomplished by cryptographic means such as a public key system. The owner of the certificate (in general the manufacturer of the measuring instrument) generates an electronic signature of the revised software or software part using the *secret key* in the manufactory. The *public key* is stored in a legally relevant software part of the measuring instrument receiving the signed revised software. The signature is checked using the *public key* when loading the revised software into the measuring instrument. If the signature of the loaded software is OK, it is installed and activated; if it fails the check, the loaded revised software is discarded, and the instrument continues to operate with the current version of the software or switches to an inoperable mode.

6.2.8.4.5 Technical means shall be employed to ensure the integrity of the loaded software, i.e. that it has not been inadmissibly changed before loading. This can be accomplished by adding a checksum or hash code of the loaded software and verifying it during the loading procedure.

6.2.8.4.6 An audit trail shall be employed to ensure that traced updates of the legally relevant software part are adequately traceable within the instrument for subsequent verification and surveillance or inspection.

The audit trail shall contain at minimum the following information:

- success/ failure of the update procedure;
- software identification of the installed version;
- software identification of the previous installed version;
- time stamp of the event;
- identification of the downloading party if available.

An entry is generated for each update attempt regardless of the success.

The storage device that supports the traced update shall have a sufficient capacity to ensure the traceability of traced updates of the legally relevant software part between at least two successive verifications of a measuring instrument in the field/inspections. After having reached the limit of the storage for the audit trail, it shall be ensured by technical means that further downloads are impossible without breaking a seal.

The audit trail shall be displayed or printed on command. The certificate shall describe how the audit trail may be displayed or printed.

Note: This requirement enables inspection authorities, which are responsible for the metrological surveillance of legally controlled instruments, to back-trace traced updates of the legally relevant software part over an adequate period of time (depending on national legislation).

6.2.8.4.7 Depending on the needs and on national legislation it may be necessary for the user or owner of the measuring instrument to give their consent to a traced update. The measuring instrument shall have a feature for the user or owner to express their consent, e.g. a push button, before the update starts. It shall be possible to enable and disable the feature, e.g. by a switch that can be sealed or by a parameter. If the feature is enabled, each traced update needs to be initiated by the user or owner. If it is disabled, no activity by the user or owner is necessary to perform a traced update.

6.2.8.4.8 If the loaded software fails the integrity test (6.2.8.4.5) or the authenticity test (6.2.8.4.4), the instrument shall discard the new version and use the previous version of the software or switch to an inoperable mode. In this mode, the measuring functions shall

be inhibited. It shall only be possible to resume the download procedure, or to show an error. If the audit trail has no more capacity (6.2.8.4.6), or the user or owner denies consent (6.2.8.4.7), the update procedure should not start at all.

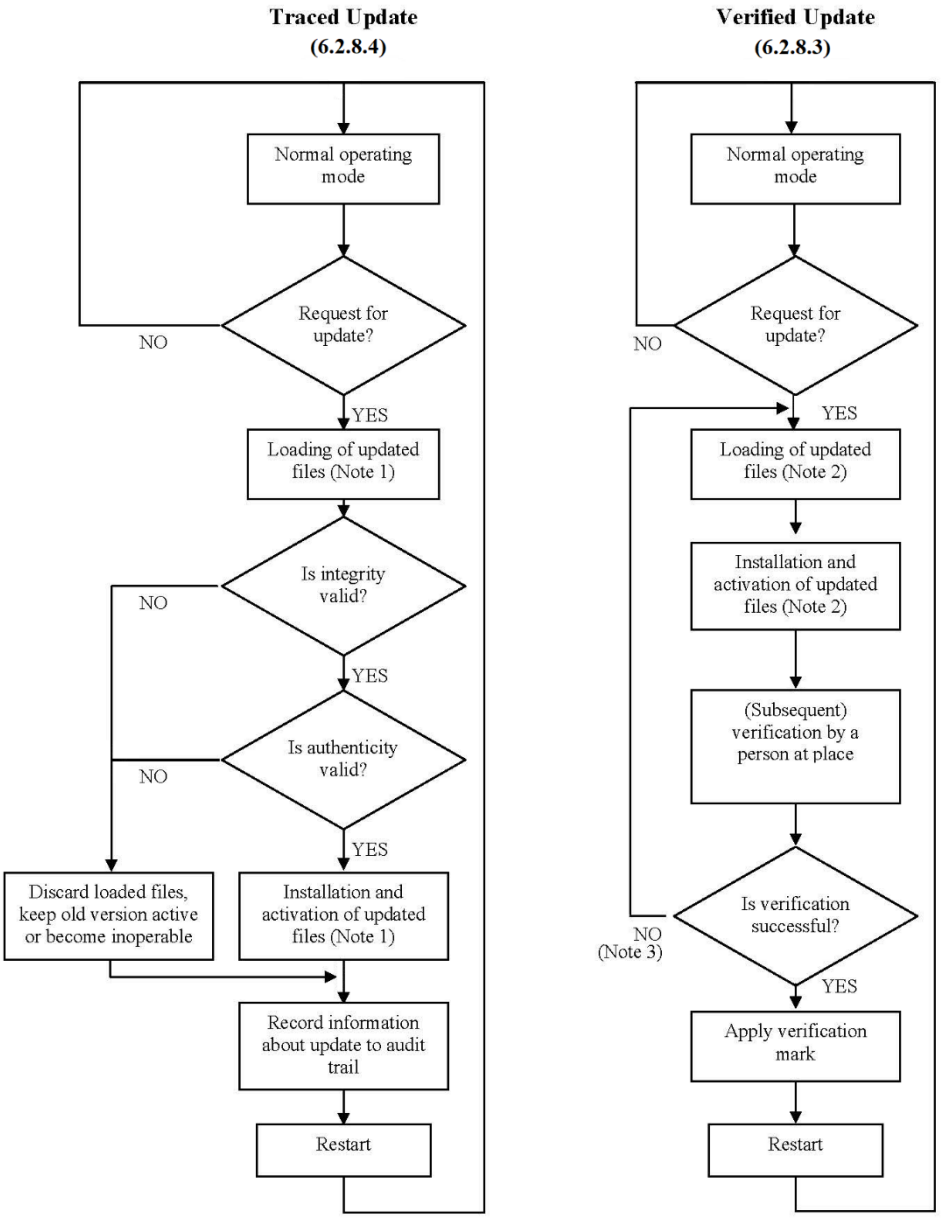

Figure 1 – Software update procedure

Note 1: In the case of a traced update, updating is separated into two steps: "loading" and "installing/activating". This implies that the software is temporarily stored after loading without being activated because it shall be possible to discard the loaded software and revert to the old version, if the checks fail.

Note 2: In the case of a verified update, the software may also be loaded and temporarily stored before installation, but depending on the technical solution loading and installation may also be accomplished in one step.

Note 3: Here, only failure of the verification of a measuring instrument due to the software update is considered. Failure due to other reasons does not require re-loading and re-installing of the software, symbolised by the NO-branch.

6.2.8.5 The relevant Recommendation may require the setting of certain device-specific parameters to be available to the user. In such a case, the measuring instrument shall be fitted with a facility to automatically and non-erasably record any adjustment of the device-specific parameter, e.g. an audit trail. The instrument shall be capable of presenting the recorded data.

Note: The audit trails are part of the legally relevant software part, see 6.1.3.2.1.

6.2.8.6 When the software is updated, the audit trail should not be erased or overwritten.

7 Type evaluation

7.1 Software documentation to be supplied for type evaluation

7.1.1 General

For type evaluation the manufacturer of the measuring instrument shall declare and document all functions, relevant data structures and software interfaces of the legally relevant software part that are implemented in the instrument. All commands and their effects shall be described completely in the software documentation to be submitted for type evaluation.

Furthermore, the application for type evaluation shall be accompanied by a document or other evidence that supports the assumption that the design and characteristics of the software of the measuring instrument comply with the requirements of the relevant Recommendation, in which the general requirements of this Document have been incorporated.

7.1.2 Contents of the documentation

The documentation (for each measuring instrument/component) shall at least include:

- description of the legally relevant software and how the requirements are met:
  - list of software modules that belong to the legally relevant part;
  - description of the software interfaces of the legally relevant software part and of the commands and data flows via this interface;
  - depending on the evaluation method chosen in the relevant Recommendation (see 7.3 and 7.4) the source code shall be made available to the type evaluation authority if raised risk level is required by the relevant Recommendation;
  - list of parameters to be protected and description of protection means;

- description of suitable system configuration and minimal required resources (see 6.2.6);
- description of the security means of the operating system (password, etc. if applicable);
- description of the (software) sealing method(s);
- overview of the system hardware, e.g. topology block diagram, type of computer(s), type of network, etc. Where a hardware component is deemed legally relevant or where it performs legally relevant functions, this should also be identified;
- description of the accuracy of the algorithms (e.g. filtering of A/D conversion results, price calculation, rounding algorithms, etc.);
- description of the user interface, menus and dialogues;
- software identification and instructions for obtaining it from an instrument in use;
- list of commands of each hardware interface of the measuring instrument/component;
- list of durability errors that are detected by the software and, if necessary for understanding, a description of the detecting algorithms;
- description of datasets stored or transmitted;
- if detection of significant defects is realised in the software, a list of significant defects that are detected and a description of the detecting algorithm;
- if fault detection is realised in the software, a list of faults that are detected and a description of the detecting algorithm;
- if an audit trail is realised in the software, a description of how to access the audit trail;
- the operating manual.

7.2 Requirements for the evaluation procedure

7.2.1 General

In the framework of type evaluation, test procedures are based on well-defined test setups and test conditions and can rely on metrologically traceable comparative measurements. The accuracy or correctness of software in general cannot be measured in a metrological sense, though there are standards that prescribe how to "measure" software quality [e.g. ISO/IEC 25040:2011 series [5]]. The procedures described here take into consideration both the legal metrology needs and also well-known evaluation and verification methods in software engineering, but which do not have the same goals (e.g. a software developer who searches for errors but who also optimises performance). As shown in 7.4, each software requirement needs individual adaptation of suitable evaluation procedures. The effort for the procedure should reflect the risk level.

The aim is to verify the fact that the instrument to be certified complies with the requirements of the relevant Recommendation. For software-controlled instruments the evaluation procedure comprises examinations, analysis, and tests and the relevant Recommendation shall include an appropriate selection of methods described below.

The methods described below focus on the type evaluation. Verifications of every single instrument in use in the field are not covered by those evaluation methods. Refer to clause 8 *Verification of a measuring instrument* for more information.

The methods specified for software evaluation are described in 7.3. Combinations of these methods forming a complete software evaluation procedure adapted to all requirements defined in clause 6 are specified in 7.4.

The manufacturer shall attest that no hidden or undocumented properties exist. (e.g. parameters, commands, functions, backdoors.)

This Document does not ask manufacturers for extra declarations that documentation is correct and complete. However, any country may require this declaration, as a part of the specified software examination process.

7.2.2 Information to be included in the certificate

The following information shall be included in the certificate:

- software identification of all certified versions;
- method to display the current software identification on the certified instrument in use;
- securing means as well as means to provide evidence of an intervention and the method to check them (e.g. hardware seals, event counters, audit trails.);
- software modules under legal control;
- if applicable:
  - means of integrity protection checking;
  - Software operating environment.

7.3 Verification and evaluation methods

7.3.1 Overview of methods and their application

The selection and sequence of the following methods are not prescribed and may vary in a software evaluation procedure from case to case.

This is a rough overview. For more details, see 7.3.2.

Table 1 – Overview of the proposed selected verification and evaluation methods

Abbreviation	Description	Application	Preconditions, tools for application	Special skills for performing
AD	Analysis of the documentation and evaluation of the design (7.3.2.1)	Always	Documentation	-
VFTM	Verification by functional testing of metrological functions (7.3.2.2)	Correctness of the algorithms, uncertainty, compensating and correcting algorithms, rules for price calculation	Documentation, specimen	-

VFTSw	Verification by functional testing of software functions (7.3.2.3)	Correct functioning of communication, indication, evidence of intervention, protection against operating errors, protection of parameters, detection of significant defects	Documentation, specimen	-
DFA	Metrological data flow analysis (7.3.2.4)	Software separation, evaluation of the impact of commands on the instrument's functions	Source code, tools for analysing source code	Knowledge of programming languages
CIWT	Code inspection and walkthrough (7.3.2.5)	All purposes	Source code, tools for analysing source code	Knowledge of programming languages
SMT	Software module testing (7.3.2.6)	All purposes when input and output can clearly be defined	Source code, testing environment	Knowledge of programming languages

Table 2 – Recommendations for combinations of evaluation and verification methods for the various software requirements (acronyms defined in Table 1)

Requirement		**Examination level A (normal examination level)**	**Examination level B (extended examination level)**	**Comment**
General requirements				
6.1.1	Software identification	AD + VFTSw	AD + VFTSw + CIWT	Select "B" if high conformity is required
6.1.2	Correctness of algorithms and functions	AD + VFTM	AD + VFTM + CIWT/SMT	
Software protection				
6.1.3.1	Prevention of misuse	AD + VFTSw	AD + VFTSw	
6.1.3.2	Evidence of intervention	AD + VFTSw	AD + VFTSw + DFA/CIWT/SMT	Select "B" in case of high risk of fraud
Support of hardware features				
6.1.4.1	Detection of significant defects	AD + VFTSw	AD + VFTSw + CIWT + SMT	Select "B" if high reliability is required
6.1.4.2	Durability protection	AD + VFTSw	AD + VFTSw + CIWT + SMT	Select "B" if high reliability is required
6.1.5	Time stamps	AD + VFTSw	AD + VFTSw + SMT	
Specification and separation of legally relevant parts and specification of interfaces				
6.2.2.1	Separation of components	AD	AD + DFA/CIWT	

Requirement		Examination level A (normal examination level)	Examination level B (extended examination level)	Comment
6.2.2.2	Specification and separation of software parts	AD	AD + DFA/CIWT	
6.2.3	Shared indications	AD + VFTM/ VFTSw	AD + VFTM/ VFTSw + DFA/CIWT	
6.2.4	Storage of data	AD + VFTSw	AD + VFTSw + CIWT/SMT	Select “B” if storage of measurement data in unsecure storages is foreseen
6.2.4.2	The measurement data stored shall be accompanied by all relevant information necessary for future legally relevant use	AD + VFTSw	AD + VFTSw + CIWT/SMT	Select “B” in case of high risk of fraud
6.2.4.3	The stored measurement data shall be protected by software means to guarantee the authenticity, integrity and, if necessary, correctness of the information concerning the time of measurement	AD + VFTSw	AD + VFTSw + SMT	
6.2.4.4	Automatic storing	AD + VFTSw	AD + VFTSw + SMT	
6.2.5	Transmission via communication lines	AD + VFTSw	AD + VFTSw + CIWT/SMT	Select “B” if transmission of measurement data in open system is foreseen
6.2.5.1	The measurement data transmitted shall be accompanied by all relevant information necessary for future legally relevant use.	AD + VFTSw	AD + VFTSw + CIWT/SMT	Select “B” in case of high risk of fraud
6.2.5.2	The transmitted data shall be protected by software means to guarantee the authenticity, integrity and, if necessary correctness of the information concerning the time of measurement.	AD + VFTSw	AD + VFTSw + SMT/	
6.2.5.3	Transmission delay or interruption	AD + VFTSw	AD + VFTSw + SMT	Select “B” in case of high risk of fraud, e.g. transmission in open systems
6.2.6	Compatibility of operating systems and hardware	AD + VFTSw	AD + VFTSw + SMT	
6.2.6.2	Hardware interfaces not equipped with a protective software interface shall not be able to inadmissibly influence the legally relevant software part.	AD + VFTSw	AD + VFTSw + SMT	

Requirement		Examination level A (normal examination level)	Examination level B (extended examination level)	Comment
6.2.6.3	If a secure boot process is needed to ensure protection of the legally relevant software part, the following requirements apply.	AD + VFTSw	AD + VFTSw + SMT	
6.2.6.4	The combination of the legally relevant software part and the operating system shall ensure that there are enough resources for the operation of the legally relevant application.	AD + VFTSw	AD + VFTSw + SMT	
6.2.6.5	Protection during use	AD + VFTSw	AD + VFTM/ VFTSw + DFA	
6.2.6.6	Communication with the legally relevant software part shall be made via protective interfaces.	AD + VFTSw	AD + VFTM/ VFTSw + DFA	
6.2.6.7	Identification and traceability	AD + VFTSw	AD + VFTSw + SMT	
6.2.6.8	The manufacturer shall identify the hardware and software environment that is suitable. Minimum resources and a suitable configuration necessary for correct functioning shall be declared by the manufacturer.	AD + VFTSw	AD + VFTSw + SMT	
6.2.6.9	Technical means shall be provided in the legally relevant software to prevent operation, if the minimum resources or a suitable configuration are not met.	AD + VFTSw	AD + VFTSw + SMT	
Maintenance and re-configuration				
6.2.8.3	Verified update	AD	AD	
6.2.8.4	Traced update	AD + VFTSw	AD + VFTSw + CIWT/SMT	Select “B” in case of high risk of fraud

7.3.2 Description of selected verification and evaluation methods

7.3.2.1 Analysis of Documentation and Specification and Evaluation of the Design (AD)

Application:

Basic procedure for software evaluation.

Preconditions:

The procedure is based on the manufacturer's documentation of the measuring instrument. This documentation shall have a scope which is adequate for the application:

1) Specification of the externally accessible functions of the instrument in a general form (suitable for simple instruments with no interfaces except a display, all features verifiable by functional testing, low risk of fraud).
2) Specification of the software functions and interfaces (necessary for instruments with interfaces and for instrument functions that cannot be functionally tested and in case of increased risk of fraud). The description shall make evident and explain all software functions that may have an impact on the metrological features.
3) Concerning interfaces, the documentation shall include a complete list of commands or signals that the software is able to interpret. The effect of each command shall be documented in detail. The way shall be described in which the instrument reacts on commands that are not described in the documentation.
4) Additional documentation of the software for complex measuring algorithms, cryptographic functions, or crucial timing constraints shall be provided, if necessary for understanding and evaluating the software functions.

A general precondition for examination is the completeness of the documentation and the clear identification of the EUT, i.e. of the software packages that contribute to the metrological functions (see 7.1.2).

Description:

The examiner evaluates the functions and features of the measuring instrument using the documentation and decides whether they comply with the requirements of the relevant Recommendation. Metrological requirements as well as software-functional requirements defined in clause 6 (e.g. evidence of intervention, protection of adjustment parameters, disallowed functions, communication with other devices, update of software, detection of significant defects, etc.) shall be considered and evaluated. This task may be supported by the Software Evaluation Report Format (see Annex B).

Result:

The procedure gives a result for all characteristics of the measuring instrument, provided that the appropriate documentation has been submitted by the manufacturer. The result should be documented in a clause related to software in a Software Evaluation Report (see Annex B) included in the Evaluation Report Format of the relevant Recommendation.

Complementary procedures:

Additional procedures should be applied, if examining the documentation cannot provide substantiated evaluation results. In most cases "Verifying the metrological functions by functional testing" (see 7.3.2.2) is a complementary procedure.

Reference:

IEC 61508-5:2010 [7].

7.3.2.2 Verification by Functional Testing of the Metrological Functions (VFTM)

Application:

For verifying correctness of algorithms for calculating the measurement result from measurement data, for linearisation of a characteristic, compensation of environmental influences, rounding in price calculation, etc.

Preconditions:

Operating manual, functioning specimen, metrological references, test equipment, test cases, instructions for test equipment.

When it is not clear how to verify a function of a software part, the onus to develop a test method should be placed on the manufacturer. In addition, the services of the programmer should be made available to the examiner for the purposes of answering questions.

Description:

Most of the evaluation and verification methods described in Recommendations are based on reference measurements under various conditions. Their application is not restricted to a certain technology of the instrument. Although it does not aim primarily at verifying the software, the test result can be interpreted as a verification of some software parts, in general even the metrologically most important. If the tests described in the relevant Recommendation cover all the metrologically relevant features of the instrument, the corresponding software parts can be regarded as being verified. In general, no additional software analysis or test needs to be applied to verify the metrological features of the measuring instrument.

Result:

Algorithms are correct or not correct. Measurement results under all conditions are within the maximum permissible error (MPE) or not.

Complementary procedures:

The method is normally an enhancement of 7.3.2.1. In certain cases, it may be easier or more effective to combine the method with examinations based on the source code (7.3.2.5) or by simulating input signals (7.3.2.6) e.g. for dynamic measurements.

References:

Various specific Recommendations.

7.3.2.3 Verification by Functional Testing of the Software Functions (VFTSw)

Application:

For evaluation of e.g. protection of parameters, indication of a software identification, software supported detection of significant defects, configuration of the system (especially of the software environment), etc.

Preconditions:

Operating manual, software documentation, functioning specimen, test equipment, test cases, instructions for test equipment.

When it is not clear how to verify a function of a software part, the onus to develop a test method should be placed on the manufacturer. In addition, the services of the programmer should be made available to the examiner for the purposes of answering questions.

Description:

Required features described in the operating manual, instrument documentation or software documentation are checked practically. If they are software-controlled, they are to be regarded as verified if they function correctly without any further software analysis. Features addressed here are e.g.:

- normal operation of the instrument, if its operation is software-controlled. All switches or keys and described combinations should be employed and the reaction of the instrument evaluated. In graphical user interfaces, all menus and other graphical elements should be activated and checked;
- effectiveness of parameter protection may be checked by activating the protection means and trying to change a parameter;
- effectiveness of the protection of stored data may be checked by changing some data in the file and then checking whether this is detected by the software;
- indication of the software identification may be verified by practical checking;
- if detection of significant defects is software supported, the relevant software parts may be verified by provoking, implementing or simulating a fault and checking the correct reaction of the instrument;
- protection means that there is evidence of an intervention if changes are made to software, parameters, audit trails, etc. This can be tested by making changes and checking if this leads to evidence of an intervention.

Result:

Software-controlled feature under consideration is acceptable or not acceptable.

Complementary procedures:

Some features or functions of a software-controlled instrument cannot be practically verified as described. If the instrument has interfaces, it is in general not possible to detect unauthorised commands only by trying commands at random. Besides that, a sender is needed to generate these commands. For the normal examination level method in 7.3.2.1 may cover this requirement. For the extended examination level, a software analysis such as 7.3.2.4 or 7.3.2.5 is necessary.

References:

WELMEC Guide 2.3, Section 3 [8]; WELMEC Guide 7.2, Sections 4.2 and 5.2[9].

7.3.2.4 Metrological Dataflow Analysis (DFA)

Application:

For analysis of the software design concerning the control of the data flow of measurement information through the data domains that are subject to legal control, including the examination of the software separation.

Preconditions:

Software documentation, source code, editor, text search program or special tools. Knowledge of programming languages.

Description:

It is the aim of this method to find all parts of the software that are involved in the calculation of the measurement result or that may have an impact on it. Starting from the hardware port where raw data from the sensor are available, the subroutine that reads them is searched. This subroutine will store them in a variable after possibly having done some processing. From this variable the intermediate value is read by another subroutine and so forth until the completed measurement result is output to the display. All variables that are used as storage for intermediate measurement data and all subroutines processing and transporting these data can be found in the source code simply by using a text editor and a text search program to find all other occurrences of the variable or the subroutine name.

Other data flows can be found by this method, e.g. from software interfaces to the interpreter of received commands. Furthermore, circumvention of a software interface (see 6.2.2.2) can be detected.

Result:

It can be verified whether software separation according to 6.2.2.2 is acceptable or not acceptable.

It can be verified whether the documented list of commands for each interface is complete or not.

Complementary procedures:

This method is recommended if software separation is realised and if high conformity or strong protection against manipulation is required. It is an enhancement to 7.3.2.1-7.3.2.3 and to 7.3.2.5.

Reference:

IEC 61131-3.

7.3.2.5 Code Inspection and Walk Through (CIWT)

Application:

Any feature of the software may be verified with this method if extended examination intensity is necessary.

Preconditions:

Source code, text editor, tools. Knowledge of programming languages.

Description:

The examiner walks through the source code assignment by assignment, evaluating the respective part of the code to determine whether the requirements are fulfilled and whether the functions and features are in compliance with the documentation.

The examiner may also concentrate on algorithms or functions that he has identified as complex, error-prone, insufficiently documented, etc. and inspect the respective part of the source code by analysing and checking.

Prior to these examination steps the examiner will have identified the legally relevant software part, e.g. by applying the metrological data flow analysis (see 7.3.2.4). In general code inspection or walk through is limited to this part.

Result:

Implementation compatible with the software documentation and in compliance with the requirements or not.

Complementary procedures:

This is an enhanced method, additional to 7.3.2.1 and 7.3.2.4. Normally it is only applied in spot checks.

Reference:

IEC 61508-5:2010 [7].

7.3.2.6 Software Module Testing (SMT)

Application:

This method is only used in exceptional cases. It is applied when functions of a software module cannot be examined exclusively on the basis of written information. It is appropriate and effective in the verification of dynamic measurement algorithms.

Preconditions:

Source code, development tools, functioning environment of the software module under test, input dataset and corresponding nominal output dataset or tools for automation. Skills in information technology, knowledge of programming languages. Co-operation with the programmer of the module under test is advisable.

Description:

The software module under test is integrated in a test environment, i.e. a specific test program that calls the module under test and provides it with all necessary input data. The test program receives actual output data from the module under test and compares them with the nominal values.

Result:

Module under test is correct or not.

Complementary procedures:

This is an enhanced method, additional to 7.3.2.2 or 7.3.2.5.

Reference:

IEC 61508-5:2010 [7].

7.4 Software evaluation procedure

The software evaluation procedure consists of a combination of evaluation and verification methods. The relevant Recommendation may specify details concerning the software evaluation procedure, including

a) which of the evaluation and verification methods described in 7.3 shall be carried out for the requirement under consideration,

b) how the evaluation of test results shall be performed,

c) which results should be included in the software test report, which results should be included in the evaluation report and which results should be integrated in the certificate (see Annex B).

In Table 2 two alternative examination levels Normal (A) and Extended (B) for the software evaluation procedures are defined. DFA, CIWT and SMT methods are only suggested for level B. Level B implies an extended examination compared to A. The selection of level B shall be justified by the PGs together with evidence of mitigated risk. A selection between A and B examination levels may be made in the relevant Recommendation – different or equal for each requirement – in accordance with the expected

- risk of fraud,
- area of application,
- required conformity to certified type, and
- risk of wrong measurement result due to operating errors.

See clause 4 for preliminary guidance on risk assessment.

7.5 Equipment Under Test (EUT)

Normally, tests are carried out on the complete measuring instrument (functional testing). If the size or configuration of the measuring instrument does not lend itself to testing as a whole unit or if only a separate component or software module of the measuring instrument is concerned, the relevant Recommendation may indicate that the tests, or certain tests, shall be carried out on the components or software modules separately, provided that, in the case of tests with the components or software modules in operation, these are included in a simulated setup, sufficiently representative of its normal operation. The applicant is responsible for the provision of all the required equipment and specimens.

8 Verification of a measuring instrument

8.1 General

If metrological control of measuring instruments is prescribed in a country, there shall be means to check in the field during operation the identity of the software, the validity of the adjustment and the conformity to the certified type.

The relevant Recommendation may require carrying out the verification of the software in one or more stages according to the nature of the considered measuring instrument.

The verification of the software shall include

- an examination of the conformity of the software to verify that it is the certified version (e.g. check of the software identification, check of securing means),
- an examination of the configuration to verify that it is compatible with the declared minimal configuration, if given in the certificate,
- an examination of the inputs/outputs of the measuring instrument to verify that they are free of unwanted side effects, and
- an examination of the device-specific parameters (especially the adjustment parameters) to verify that they are correctly set.

PGs shall consider the following subclause when writing instrument-specific verification procedures. The methods given in 8.2 are proposed as the standard procedure.

8.2 Verification methods, test items

The following methods comprise the verification steps which are needed to check the requirements of 6.1 and 6.2. The aspects in 8.2.1 to 8.2.4 shall be examined by the instructions listedin the corresponding subclause below.

8.2.1 Documents

The initial step of any software verification consists of checking the EUT for compliance with the certificate and its annexes:

- check whether the certificate is valid;
- check whether the EUT complies with the pattern as described in the certificate and its annexes;
- check whether the operating manual is available (if required).

8.2.2 Integrity of the software

Software integrity may be checked in one of two ways:

- indirectly: Check whether all seals required in the certificate are set at the right place and are intact;
- directly: Check the software identifiers as required in the certificate.

Note: The second item overlaps with the first item of 8.2.4.

Example:

Calculation of a checksum of the program code that is compared with the nominal value.

8.2.3 Parameters

8.2.3.1 Correctness

The correctness of parameters may be checked as follows:

- indirect metrological verification of parameters: Perform a measurement and compare the results with a reference;
- check whether all settable parameters are within the allowed range.

8.2.3.2 Integrity

The integrity of parameters may be checked as follows:

- check whether the seals protecting the parameters are intact;
- check the audit trail or log for entries concerning parameters.

8.2.4 Identity of the software

The identity of the software may be checked as follows:

- check that the software identifier provided by the EUT is specified as valid for use in the certificate;
- check the entries of the audit trail for traced updates (see 6.2.8.4.6).

Note: The first item overlaps with the second item of 8.2.2.

Annex A
Bibliography
(Informative)

At the time of publication, the editions indicated were valid. All referred documents are subject to revision, and the users of this Document are encouraged to investigate the possibility of applying the most recent editions of the referred documents indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

The actual status of the Standards referred to can also be found on the Internet:

IEC Publications: http://www.iec.ch/searchpub/cur_fut.htm

ISO Publications: http://www.iso.org

OIML Publications: https://www.oiml.org/en/publications/

(with free download of PDF files).

In order to avoid any misunderstanding, it is highly recommended that all references to Standards in International Recommendations and International Documents be followed by the version referred to (generally the year or date).

Ref.	Standards and reference documents	Description
[1]	OIML V 2-200:2012 International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM), 3rd Edition	Vocabulary, developed by the Joint Committee for Guides in Metrology (JCGM).
[2]	OIML D 11:2013 General requirements for measuring instruments – Environmental conditions	Guidance for establishing appropriate metrological performance testing requirements for influence quantities that may affect the measuring instruments covered by OIML Recommendations (EMC, climatic, mechanical influences).
[3]	ISO/IEC 9594-8:2017 Information technology -- Open Systems Interconnection -- The Directory: Part 8: Public-key and attribute certificate frameworks	ISO/IEC 9594-8:2017 specifies frameworks and a number of data objects that can be used to authenticate and secure the communication between two entities, e.g. between two directory service entities or between a web browser and a web server. The data objects can also be used to prove the source and integrity of data structures such as digitally signed documents.
[4]	ISO/IEC 2382-9:2015 Information technology -- Vocabulary -- Part 9: Data communication	Intended to facilitate international communication in data communication. Presents terms and definitions of selected concepts relevant to the field of data communication and identifies relationships among the entries.
[5]	ISO/IEC 25040:2011series Information technology -- Software product evaluation	The ISO/IEC 25040:2011 series of Standards gives methods for measurement, assessment and evaluation of software product quality. They describe neither methods for evaluating software production processes nor methods for cost prediction (software product quality measurements may, of course, be used for both these purposes).

Ref.	Standards and reference documents	Description
[6]	OIML V 1:2013 International vocabulary of terms in legal metrology (VIML)	The VIML includes only the concepts used in the field of legal metrology. These concepts concern the activities of the legal metrology service, the relevant documents, as well as other problems linked with this activity. Also included in this Vocabulary are certain concepts of a general character which have been drawn from the VIM.
[7]	IEC 61508-5:2010 Functional safety of electrical/ electronic/ programmable electronic safety-related systems – Part 5: Examples of methods for the determination of safety integrity levels	Provides information on the underlying concepts of risk and the relationship of risk to safety integrity (see Annex A); a number of methods that will enable the safety integrity levels for the E/E/PE safety-related systems, other technology safety-related systems and external risk reduction facilities to be determined (see Annexes, B, C, D and E). Intended for use by Technical Committees in the preparation of Standards in accordance with the principles contained in IEC Guide 104 and ISO/IEC Guide 51.
[8]	WELMEC Guide 2.3, May 2005 Issue 3 Guide for Examining Software (Weighing Instruments)	This guideline specifies basic requirements to be applied to software for free programmable, PC-based modules or peripheral devices which are linked to, or form part of, NAWIs subject to legal control.
[9]	WELMEC Guide 7.2, Issue 2018 Software Guide (Measuring Instruments Directive 2014/32/EU)	This document provides guidance to all those concerned with the application of the Measuring Instruments Directive (European Directive 2014/32/EU; MID), especially for software-equipped measuring instruments. It addresses both manufacturers of measuring instruments and notified bodies which are responsible for conformity assessment of MID instruments. By following the Guide, compliance with the software- related requirements contained in the MID can be assumed.
[10]	ISO/IEC 27005:2018 Information technology -- Security techniques -- Information security risk management	This document provides guidelines for information security risk management. This document supports the general concepts specified in ISO/IEC 27001 and is designed to assist the satisfactory implementation of information security based on a risk management approach. Knowledge of the concepts, models, processes and terminologies described in ISO/IEC 27001 and ISO/IEC 27002 is important for a complete understanding of this document. This document is applicable to all types of organisations (e.g. commercial enterprises, government agencies, non-profit organisations) which intend to manage risks that can compromise the organisation's information security.

Annex B
Example of a software test report
(Informative)

Note: The Technical Committees and Subcommittees developing OIML Recommendations should decide which information shall be included in Software Test Report, Evaluation Report and OIML Certificate of Conformity. E.g. the name, version and checksum of the executable code from the following example should be included in the Certificate.

Software Test report no XYZ122344

Evaluation of Software of the flow meter Tournesol Metering **model** TT100

The software of the measuring instrument was verified to show conformance with the requirements of OIML Recommendation R xyz.

The evaluation was based on OIML International Document D 31:2019, where the essential requirements for software are interpreted and explained. This report describes the evaluation of software needed to state conformance with the R-xyz.

Manufacturer	Applicant
Tournesol Metering	New Company
P.O. Box 1120333	Nova Street 123
100 Klow	1000 Las Dopicos
Syldavie	San Theodorod
Reference: Mr. Tryphon Tournesol	Reference: Archibald Haddock

Test object

The Tournesol Metering meter TT100 is a measuring instrument intended to measure flow in liquids. The intended range is from 1 L/s up to 2000 L/s. The basic functions of the instrument are:

- measuring of flow in liquids;
- indication of measured volume;
- interface to transducer.

The flow meter is described as a built-for-purpose device (an embedded system) with a storage device containing legally relevant data.

The flow meter TT100 is an independent instrument with a transducer connected. The transducer incorporates a temperature compensation. Adjustment of flow rates is possible by calibration parameters stored in a non-volatile memory of the transducer. It is fixed to the instrument and cannot be disconnected. The measured volume is indicated on a display. No communication with other devices is possible.

The embedded software of the measuring instrument was developed by

Tournesol Metering, P.O. Box 1120333, 100 Klow, Syldavie.

The file name of the executable code is "**tt100_12.exe**".

The verified version of this software is **V1.2c**. The software version is presented on the display upon instrument start-up and by pressing the "level" button for 4 seconds.

The source code comprises the following legally relevant files:

- main.c 12301 byte 23 Nov 2003;
- int.c 6509 byte 23 Nov 2003;
- filter.c 10897 byte 20 Oct 2003;
- input.c 2004 byte 20 Oct 2003;
- display.c 32000 byte 23 Nov 2003;
- ethernet.c 23455 byte 15 June 2002;
- driver.c 11670 byte 15 June 2002;
- calculate.c 6788 byte 23 Nov 2003.

The executable code "**tt100_12.exe**" is protected against modification by a checksum. The value of the checksum by algorithm **XYZ** is **1A2B3C**.

The evaluation was supported by the following documents from the manufacturer:

- TT 100 User Manual Release 1.6;
- TT 100 Maintenance Manual Release 1.1;
- Software description TT100 (internal design document, dated 22 Nov 2003);
- Electronic circuit diagram TT100 (drawing no 222-31, dated 15 Oct 2003).

The final version of the test object was delivered to the National Testing & Measurement Laboratory on 25 November 2003.

Results of evaluation

The evaluation was performed according to the OIML D 31:YYYY. The evaluation was performed between 1 November and 23 December 2003. A design review was held on 3 December by Dr. K. Fehler at Tournesol Metering head office in Klow. Other evaluation work was carried out at the National Testing & Measurement Laboratory by Dr. K. Fehler and Mr. S. Problème.

The following requirements were verified:

- software identification;
- correctness of algorithms and functions;
- software protection;
- prevention against accidental misuse;
- evidence of intervention;
- support of hardware features;
- storage of data, transmission via communication systems.

The following evaluation and verification methods were applied:

- analysis of the documentation and evaluation of the design;
- verification by functional testing of metrological features;
- walkthrough, code inspection;
- software module testing of module calculate.c with SDK XXX.

Result

The following requirements of the OIML D 31:YYYY were verified without any non-conformities being found:

6.1.1, 6.1.2, 6.1.3, 6.1.4, 6.2.4 and 6.2.5.

The result applies to the tested item with Serial No. 1188093-B-2004 only.

Conclusion

The software of the **Tournesol Metering TT100 V1.2c** fulfils the requirements of OIML R xyz.

National Testing & Measurement Lab.

Software Department

Signature(s):

Dr. K.E.I.N. Fehler	Mr. S.A.N.S. Problème
Technical manager	Technical Officer

Clause	Requirement	Passed	Failed	Remarks
6.1 **6.1.1**	**General requirements** **Software identification** Software of a measuring instrument/component shall be clearly identified.			
6.1.2	**Correctness of algorithms and functions** The measuring algorithms and functions of a measuring instrument shall be appropriate and functionally correct for the given application and device type.			
6.1.2 **6.1.3.1**	**Software protection** Prevention of misuse A measuring instrument shall be constructed in such a way that possibilities for unintentional, accidental, or intentional misuse are minimal.			
6.1.3.2 **6.1.3.2.1** **6.1.3.2.2** **6.1.3.2.3** **6.1.3.2.4**	**Evidence of an intervention** Software shall be protected in such a way that evidence of any intervention (e.g. software updates, parameter changes) shall be available. Software shall be secured against unauthorised modification, loading, or changes by swapping the memory device. Only clearly documented functions may be activated by the user interface, which do not influence the metrological characteristics of the instrument. Parameters that fix the legally relevant characteristics of the measuring instrument shall be secured against unauthorised modification. If necessary for the purpose of verification of a measuring instrument, displaying or printing of the current parameter settings shall be possible. Software protection shall comprise appropriate sealing by mechanical, electronic and/or cryptographic means, making an intervention impossible or evident.			
6.1.4 **6.1.4.1**	**Support of hardware features** Detection of significant defects The manufacturer of the instrument shall be required to design checking facilities into the software or hardware parts or provide means by which the hardware parts can be supported by the software parts of the instrument.			
6.1.4.2	**Durability protection** It is the manufacturer's choice to realise durability protection facilities in software or hardware, or to allow hardware facilities to be supported by software.			
6.1.5	**Time stamps** The time stamp shall be read from the clock of the instrument. Appropriate protection means shall be taken according to the risk level to be applied.			
6.2 **6.2.2**	**Requirements specific for configurations** **Specification and separation of legally relevant parts and specification of interfaces** Legally relevant parts of a measuring instrument shall not be inadmissibly influenced by other parts of the measuring instrument.			
6.2.2.1 **6.2.2.1.1** **6.2.2.1.2**	**Separation of components** Components of a measuring instrument that perform legally relevant functions shall be identified, clearly defined, and documented. It shall be demonstrated that the functions and data of legally relevant components cannot be inadmissibly influenced by commands received via the interface to the other, legally non-relevant parts.			

Clause	Requirement	Passed	Failed	Remarks
6.2.2.2 **6.2.2.2.1** **6.2.2.2.2**	**Specification and separation of software parts** The conformity requirement applies to the legally relevant software part of a measuring instrument (see 6.2.6) and it shall be made identifiable as described in 6.1.1. If the legally relevant software part communicates with other software parts, a software interface shall be defined. All communication shall be performed exclusively via this interface. The legally relevant software part and the interface shall be clearly documented. All legally relevant functions and data domains of the software shall be described to enable a type evaluation authority to decide on correct software separation.			
6.2.2.2.3 **6.2.2.2.4**	There shall be an unambiguous assignment of each command to all initiated functions or data changes in the legally relevant software part. Functions that are triggered through the software interface shall be declared and documented. Only documented functions shall be activated through the software interface. Where the legally relevant software part has been separated from the non-relevant software part, the legally relevant software part shall have priority using the resources over non-relevant software. The legally relevant process shall not be inadmissibly interrupted by legally non-relevant software.			
6.2.3	**Shared indications** If a display or printout is used both for legally relevant and legally non-relevant outputs, the legally relevant information should always be readable, and clearly distinguishable from other information.			
6.2.4 **6.2.4.2**	**Storage of data** The measurement data stored shall be accompanied by all relevant information necessary for future legally relevant use.			
6.2.4.3	The stored measurement data shall be protected by software means to guarantee authenticity, integrity and, if necessary, correctness of the information concerning the time of measurement. The software that displays or further processes the measurement data and accompanying data or the measurement result shall check the time of measurement, authenticity, and integrity of the data after having read them from the storage. If an irregularity is detected, the data shall be discarded or marked unusable.			
6.2.4.4 **6.2.4.4.1** **6.2.4.4.2**	**Automatic storing** When data storage is required, measurement data shall be stored automatically when the measurement is concluded. The storage device shall have sufficient permanency to ensure that the measurement data are not corrupted under normal storage conditions. There shall be sufficient memory storage for the intended application. Stored data may be deleted if either: - the transaction is settled; or - these data are printed by a printing device subject to legal control.			
6.2.5 **6.2.5.1**	**Transmission via communication lines** The measurement data transmitted shall be accompanied by all relevant information necessary for future legally relevant use.			

Clause	Requirement	Passed	Failed	Remarks
6.2.5.2	The transmitted data shall be protected by software means to guarantee the authenticity, integrity and, if necessary, correctness of the information concerning the time of measurement. The software that displays or further processes the measurement data and accompanying data shall check the time of measurement, authenticity, and integrity of the data received from a transmission channel. If an irregularity is detected, the data shall be discarded or marked unusable.			
6.2.5.3	**Transmission delay or interruption** The measurement shall not be inadmissibly influenced by a transmission delay or interruption. If network services become unavailable or very slow, no measurement data shall be lost.			
6.2.6 **6.2.6.2**	**Compatibility of operating systems and hardware** Hardware interfaces not equipped with a protective software interface shall not be able to inadmissibly influence the legally relevant software part.			
6.2.6.3 **6.2.6.3.1** **6.2.6.3.2** **6.2.6.3.3** **6.2.6.3.4** **6.2.6.3.5**	**Boot process** If a secure boot process is needed to ensure protection of the legally relevant software part, the following requirements apply. In order to ensure integrity and authenticity of the legally relevant software part, a chain of trust shall be established over the individual components of the boot process. The processing of the chain of trust can be interrupted, as long as its integrity is preserved. The boot configuration shall be protected against modifications. Booting via open interfaces shall be prohibited.			
6.2.6.4	**System resources** The combination of the legally relevant software part and the operating system shall ensure that there are enough resources for the operation of the legally relevant application.			
6.2.6.5 **6.2.6.5.1** **6.2.6.5.2** **6.2.6.5.3** **6.2.6.5.4**	**Protection during use** The operation of software that is not legally relevant shall not inadmissibly influence the legally relevant application. The combination of the legally relevant software part and the operating system shall ensure that the legally relevant display is distinguishable. The access control shall be configured in such way that the intended use cannot be inadmissibly influenced. The administration tasks of the legally relevant software part and of the legally relevant part of the operating system shall be protected.			
6.2.6.6	**Communication with the legally relevant software part** Communication with the legally relevant software part shall be made via protective interfaces.			
6.2.6.7 **6.2.6.7.1** **6.2.6.7.2**	**Identification and traceability** The configuration of the operating system shall be identifiable. The identifier shall be displayed by the measuring instrument on command or during operation. Configuration settings of the operating system shall be protected in such a way that evidence of an intervention is available.			
6.2.6.8	The manufacturer shall identify the hardware and software environment that is suitable. Minimum resources and a suitable configuration necessary for correct functioning shall be declared by the manufacturer.			

Clause	Requirement	Passed	Failed	Remarks
6.2.6.9	Technical means shall be provided to prevent operation, if the minimum resources or a suitable configuration are not met.			
6.2.8 **6.2.8.2**	**Maintenance and reconfiguration** Only versions of the legally relevant software part that conform to the certified type are allowed for use.			
6.2.8.3	**Verified update** After the update of the legally relevant software of a measuring instrument (exchange with another certified version or re-installation) the measuring instrument should not be employed for legal purposes before a verification of the measuring instrument has been performed and the securing means have been renewed and the protection means have been reactivated.			
6.2.8.4 **6.2.8.4.2** **6.2.8.4.3** **6.2.8.4.4** **6.2.8.4.5** **6.2.8.4.6** **6.2.8.4.7** **6.2.8.4.8**	**Traced update** Traced update of software shall be automatic. If some of the securing or protection measures of the instrument are turned off to enable updating, they shall be turned on again immediately after update, independent of the result of the update process. Software shall be protected in such a way that evidence of any intervention shall be available. During an update, any existing audit trail information and event counter value shall be retained. Technical means shall be employed to guarantee the authenticity of the loaded software. Technical means shall be employed to ensure the integrity of the loaded software, i.e. that it has not been inadmissibly changed before loading. An audit trail shall be employed to ensure that traced updates of the legally relevant software part are adequately traceable within the instrument. Depending on the needs and national legislation it may be necessary for the user or owner of the measuring instrument to give their consent. If the loaded software fails the integrity test or the authenticity test, the instrument shall discard the new version and use the previous version of the software or switch to an inoperable mode.			
6.2.8.5	The measuring instrument shall be fitted with a facility to automatically and non-erasably record any adjustment of the device specific parameter, e.g. an audit trail. The instrument shall be capable of presenting the recorded data.			
6.2.8.6	Audit trails should not be exchanged when the software is updated.			

Annex C
Remarks on measurement terminology
(Informative)

Note: This informative Annex is intended to illustrate the terms and definitions related to the measurement process and their usage in this OIML Document.

In general, this OIML Document distinguishes between measurement data and measurement metadata. If both are used together, measurement data is put into context and data plus metadata then become measurement information. Since the definition of measurement result includes the measured quantity value attributed to the measurand (here symbolised for easy association with the following figures by the symbol ■), with other additional relevant information, the measurement result needs to include measurement result relevant data (symbol ♣) and measurement result relevant metadata (symbol ♠) as well as the measured quantity value. The measured quantity value attributed to the measurand itself also consists of data and metadata, which qualify as measurement data and measurement metadata respectively.

Nevertheless, there may be other measurement data and metadata that are relevant for the measurement process but which do not become part of the measurement result. To distinguish between both usages of data and metadata, this OIML Document uses the derived concepts measurement result relevant data ♣, measurement result relevant metadata ♠, measurement process data ♥ and measurement process metadata ♦, see Figure A.1.

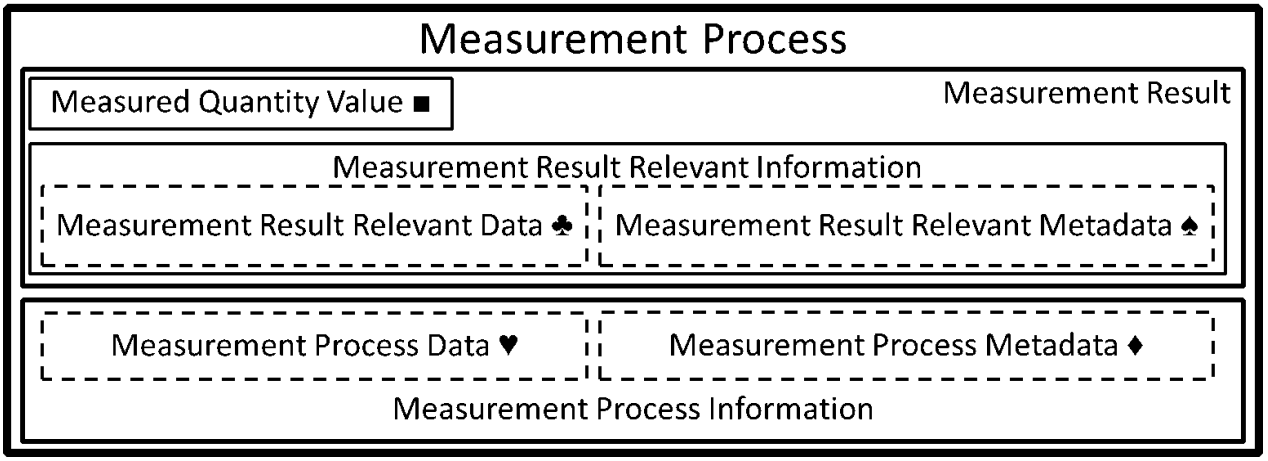

Figure A.1 – Visual representation of measurement process and measurement result, regarding their corresponding relevant data and metadata

Measurement data, therefore, includes measurement result relevant data ♣ and measurement process data ♥, whereas measurement metadata includes measurement result relevant metadata ♠ and measurement process metadata ♦.

Figure A.2 contains a flowchart to illustrate the distinction between the terms relevant to the result or relevant to the measurement process, used to describe measurements.

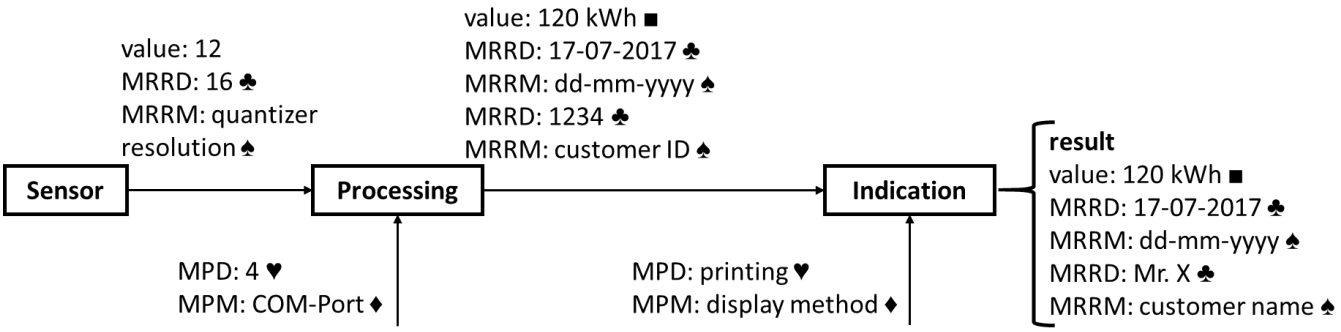

Figure A.2 – Flowchart of a measurement process, giving examples for the different terms.

Figure A.2 shows a simple example of a measurement process. For each logical step (from data acquisition by the sensor to indication of the result), the measured quantity value (■), the measurement result relevant data (MRRD, ♣) and the measurement result relevant metadata (MRRM, ♠), as well as the measurement process data (MPD, ♥) and the measurement process metadata (MPM, ♦), are given as examples.

When the sensor delivers a raw value of 12, additional measurement information relevant to the result might be the ADC's quantiser resolution of 16 bits, where 16 is the data ♣ and "quantiser resolution" is the metadata ♠ needed to interpret the data ♣. Similarly, during processing, the value could be transformed into a measured quantity value ■ (value + unit) with the additional measurement result relevant data ♣ 17-07-2017 as a time stamp, whose format (day-month-year) could be seen as metadata ♠. In both cases, the measurement data ♣ is needed to process the value and to form part of the result while the metadata ♠ is needed for the correct interpretation of the data ♣.

Another strand of measurement information (data + metadata) is related to the actual measurement process: For acquisition of the raw value, the COM-Port number 4 might be used, where 4 is the measurement process data ♥ and the measurement process metadata ♦ "COM-Port" helps to understand the data ♥ element. In the given example, indication of the result can be by means of a display or by printing. Moreover, the process data ♥ "printing" with the needed process metadata ♦ "display method" are necessary for the measurement process, but they will not become part of the result. This demonstrates that there are at least two distinct information flows within this measurement process, where both might be legally relevant.

Under certain circumstances, measurement process data ♥ might become measurement result relevant data ♣. In the given example, the COM-Port number 4 links the measured quantity value ■ to a customer ID 1234, thus turning measurement process data ♥ into measurement result relevant data ♣ during the processing step.

Annex D
Index

Acceptable solution: 6.1.1.

Audit trail: 3.1.1; 3.1.46; 6.1.3.2.1; 6.1.3.2.4; 6.2.6.5.4; 6.2.6.7.2; 6.2.8.4.1; 6.2.8.4.3; 6.2.8.4.6; 6.2.8.4.8; 6.2.8.5; 6.2.8.6; 7.1.2; 7.2.2; 7.3.2.3; 8.2.3.2; 8.2.4.

Authentication: 3.1.2; 3.1.3; 6.2.8.4.1.

Authenticity: 3.1.3; 3.1.8; 3.1.13; 6.1.3.2.4; 6.2.4.3; 6.2.5.2; 6.2.6.3.2; 6.2.8.4.4; 6.2.8.4.8.

Checking facility: 3.1.5; 6.1.4.1.

Commands: 3.1.49; 6.1.3.2.2; 6.2.2.1.2; 6.2.2.2.2; 6.2.2.2.3; 6.2.6.6 7.1.1; 7.1.2; 7.2.1; 7.3.1; 7.3.2.1; 7.3.2.3; 7.3.2.4.

Communication: 3.1.6; 3.1.56; 5.2; 6.1.1; 6.2.2.2.2; 6.2.2.2.3; 6.2.2.2.4; 6.2.5; 6.2.6.6; 6.2.6.8; 7.3.1; 7.3.2.1.

Communication interface: 3.1.6; 6.1.1.

Cryptographic certificate: 3.1.7; 3.1.13; 6.1.3.2.4.

Cryptographic means: 3.1.8; 6.1.3.2.4; 6.2.8.4.4.

Data domain: 3.1.9; 3.1.49; 3.1.50; 3.1.51; 6.2.2.2.2; 6.2.4.4.1; 7.3.2.4.

Device-specific parameter: 3.1.10; 3.1.26; 6.1.3.2.3; 6.2.8.2; 6.2.8.5; 8.1.

Durability: 3.1.11; 6.1.4.2; 7.1.2; 7.3.1.

Electronic measuring instrument: 3.1.12; 3.1.19.

Electronic Signature: 3.1.8; 3.1.13; 6.1.3.2.4; 6.2.4.3; 6.2.5.2; 6.2.8.4.4.

Error (of indication): 3.1.14; 3.1.19; 3.1.24.

Error log: 3.1.15; 6.1.4.1.

Evaluation: 3.1.57; 3.1.58; 3.1.62; 6.1.3.2.3; 6.1.4.1; 6.2.2.1.1; 6.2.2.2.2; 7.1.1; 7.1.2; 7.2.1; 7.3.1; 7.3.2.1; 7.3.2.2; 7.3.2.3; 7.4.

Event: 3.1.1; 3.1.16; 3.1.17; 3.1.46; 3.1.55; 6.1.3.2.4; 6.2.2.2.4; 6.2.8.4.6.

Event counter: 3.1.17; 6.1.3.2.4; 6.2.8.4.3; 7.2.2.

Executable code: 3.1.18; 3.1.53; 6.1.1.

Fault: 3.1.19; 3.1.46; 7.1.2; 7.3.2.3.

Hash function: 3.1.20; 6.1.4.1.

Integrity (of programs, data, or parameters): 3.1.8; 3.1.13; 3.1.21; 6.2.4.3; 6.2.5.2; 6.2.6.3.2; 6.2.6.3.3; 6.2.8.4.1; 6.2.8.4.5; 6.2.8.4.8; 7.2.2; 7.3.1; 8.2.2; 8.2.3.2.

Interface: 3.1.4; 3.1.22; 3.1.59; 6.1.1; 6.2.2; 6.2.2.1.1; 6.2.2.1.2; 6.2.2.2.4; 6.2.6.2; 6.2.6.3.5; 7.1.2; 7.3.1; 7.3.2.1; 7.3.2.3.

Intrinsic error: 3.1.19; 3.1.24.

Legally relevant: 2.1; 3.1.1; 3.1.10; 3.1.25; 3.1.26; 3.1.27; 3.1.37; 3.1.43; 3.1.49; 3.1.52; 3.1.54; 3.1.58; 6.1.3.1; 6.1.3.2.1; 6.1.3.2.2; 6.1.3.2.3; 6.1.3.2.4; 6.1.4.1; 6.1.5; 6.2.2; 6.2.2.1.1; 6.2.2.1.2; 6.2.2.2.1; 6.2.2.2.2; 6.2.2.2.3; 6.2.2.2.4; 6.2.3; 6.2.4.2; 6.2.4.3; 6.2.4.4.1; 6.2.5.1; 6.2.5.2; 6.2.6.1; 6.2.6.2; 6.2.6.3.1; 6.2.6.3.2; 6.2.6.3.5; 6.2.6.4; 6.2.6.5.1; 6.2.6.5.2; 6.2.6.5.4; 6.2.6.6; 6.2.6.7.1; 6.2.6.9; 6.2.7; 6.2.8.1; 6.2.8.2; 6.2.8.3; 6.2.8.4.3; 6.2.8.4.4; 6.2.8.4.6; 6.2.8.5; 7.1.1; 7.1.2; 7.3.1; 7.3.2.5.

Legally relevant parameter: 3.1.10; 3.1.26; 3.1.58; 6.1.3.2.4; 6.1.4.1.

Legally relevant software: 2.1; 3.1.27; 3.1.37; 3.1.43; 3.1.52; 3.1.58; 6.1.3.2.1; 6.1.3.2.2; 6.1.4.1; 6.1.5; 6.2.2; 6.2.2.2.1; 6.2.2.2.2; 6.2.2.2.3; 6.2.2.2.4; 6.2.3; 6.2.4.3; 6.2.5.2; 6.2.6.2; 6.2.6.3.1; 6.2.6.3.2;.6.2.6.4; 6.2.6.5.1; 6.2.6.5.4; 6.2.6.6; 6.2.6.9; 6.2.7; 6.2.8.1; 6.2.8.2; 6.2.8.3; 6.2.8.4.3; 6.2.8.4.4; 6.2.8.4.6; 6.2.8.5; 7.1.1; 7.1.2; 7.3.1; 7.3.2.5.

Maximum permissible error: 3.1.28; 3.2; 7.3.2.2.

Measuring instrument: 1; 2.1; 2.2; 2.3; 3; 3.1.1; 3.1.2; 3.1.5; 3.1.6; 3.1.7; 3.1.10; 3.1.11; 3.1.12; 3.1.15; 3.1.16; 3.1.18; 3.1.19; 3.1.26; 3.1.27; 3.1.28; 3.1.29; 3.1.38; 3.1.39; 3.1.44; 3.1.46; 3.1.50; 3.1.51; 3.1.52; 3.1.57; 3.1.58; 3.1.60; 3.1.62; 4.3; 5.1; 6.1; 6.1.1; 6.1.2; 6.1.3.1; 6.1.3.2.1; 6.1.3.2.3; 6.1.3.2.4; 6.1.4.2; 6.1.5; 6.2.1; 6.2.2; 6.2.2.1.1; 6.2.2.2.1; 6.2.2.2.4; 6.2.3; 6.2.4.2; 6.2.4.3; 6.2.5.1; 6.2.5.2; 6.2.6.1; 6.2.6.7.1; 6.2.8.1; 6.2.8.2; 6.2.8.3; 6.2.8.4.1; 6.2.8.4.2; 6.2.8.4.3; 6.2.8.4.4; 6.2.8.4.6; 6.2.8.4.7; 6.2.8.4.8; 6.2.8.5; 7.1.1; 7.1.2;7.2.1; 7.3.2.1; 7.3.2.2; 7.5; 8.1.

Non-interruptible/interruptible measurement: 3.1.23; 3.1.42; 6.1.4.1.

Operating system: 3.1.4; 3.1.59; 6.1.3.2.1; 6.2.2.2.3; 6.2.2.2.4; 6.2.3; 6.2.6.1; 6.2.6.2; 6.2.6.3.5; 6.2.6.4; 6.2.6.5.2; 6.2.6.6; 6.2.6.7.1; 6.2.6.7.2; 6.2.6.8; 6.2.6.9; 7.1.2.

Performance: 3.1.11; 7.2.1.

Program code:3.1.49; 6.1.4.1; 6.2.2.2.2; 6.2.4.3; 6.2.5.2; 8.2.2.

Protective interface: 3.1.43; 6.2.6.1; 6.2.6.6.

Sealing: 3.1.44; 5.2; 6.1.3.2.1; 6.1.3.2.4; 7.1.2.

Securing: 3.1.13; 3.1.45; 6.2.2.1.1; 6.2.2.1.2; 6.2.3; 6.2.8.3; 6.2.8.4.2; 7.2.2; 8.1.

Software examination: 3.1.47; 6.1.2; 7.2.1.

Software identification: 3.1.48; 6.1.1; 6.2.8.4.6; 7.1.2; 7.2.2; 7.3.1; 7.3.2.3; 8.1.

Software interface:3.1.49; 3.1.52; 6.2.2.2.2; 6.2.2.2.3; 6.2.6.2; 6.2.6.6; 7.1.1 7.1.2; 7.3.1 7.3.2.4.

Software module: 3.1.9; 3.1.16; 3.1.27; 3.1.43; 3.1.48; 3.1.49; 3.1.50; 3.1.60; 6.1.3.2.2; 6.2.2.2.1; 6.2.4.3; 6.2.5.2; 6.2.6.6; 7.1.2; 7.2.2; 7.3.1; 7.3.2.6; 7.5.

Software protection: 3.1.51; 6.1.3; 6.1.3.2.4; 7.3.1.

Software separation: 3.1.52; 6.2.2.2.2; 6.2.2.2.4; 7.3.1; 7.3.2.4.

Source code: 3.1.53; 7.1.2;7.3.1; 7.3.2.2; 7.3.2.4; 7.3.2.5; 7.3.2.6.

Storage device: 3.1.54; 6.2.4.4.1; 6.2.8.4.6.

Test:3.2; 5.1; 6.1.2; 6.1.5; 6.2.8.4.8; 7.2.1; 7.3.1; 7.3.2.1; 7.3.2.2; 7.3.2.3; 7.3.2.6; 7.4; 7.5; 8.2.

Time stamp: 3.1.1; 3.1.55; 6.1.5; 6.2.2.1.2; 6.2.4.2; 6.2.5.1; 6.2.6.7.2; 6.2.8.4.6; 7.3.1; 7.4.

Transmission of measurement data: 3.1.56; 6.1.3.2.1; 6.2.2.1.1; 6.2.5; 6.2.5.1; 6.2.5.2; 6.2.5.3; 7.3.1.

Type-specific parameter: 3.1.26; 3.1.58; 6.1.3.2.3.

Universal device: 3.1.59; 5.2; 6.1.3.2.1; 6.2.2.1.1; 6.2.2.2.3; 6.2.2.2.4; 6.2.6.9.

User interface: 3.1.60; 6.1.1; 6.1.3.2.2; 6.2.3; 7.1.2; 7.3.2.3.

Verification: 3.1.61; 3.1.62; 6.1.3.2.3; 6.2.8.1; 6.2.8.2; 6.2.8.3; 6.2.8.4.16.2.8.4; 6.2.8.4.6; 6.2.8.4.8; 7.2.1;7.3.1; 7.3.2.2; 7.3.2.3; 7.3.2.6; 7.4; 8.1; 8.2; 8.2.1; 8.2.3.1.