NATIONAL TYPE EVALUATION PROGRAM
Software

Technical Policy · Checklists · Test Procedures

NCWM
Publication 14

National Conference on Weights and Measures

National Type Evaluation Program

Software Technical Policy

NCWM

Publication 14

© 2023

Acknowledgements

Amendments

Section Number	Amendment/Change	Page	Source
All	1st edition		2022 Software Sector
Document	Please note that the Weighing Devices publication has been thoroughly reviewed by NCWM staff. Changes have been made, but none are to change intent of the policies, checklists or test procedures, thus considered editorial. Issues or concerns should be brought to the attention of NCWM staff.	Document	Editorial

Table of Contents

National Type Evaluation Program
Software Technical Policy

1. Definitions

1.1. National Type Evaluation Program (NTEP)

A program administered by National Conference on Weights and Measures, Inc. (NCWM) in cooperation with the National Institute of Standards and Technology (NIST), state and local governments and the private sector for determining, on a uniform basis, conformance of a type, with the relevant provisions of:

- *NCWM Publication 14, National Type Evaluation Program (NTEP) Technical Policy, Checklists and Test Procedures*

- *NIST Handbook 44, Specifications, Tolerances and Other Technical Requirements for Weighing and Measuring Devices*

1.2. Type Evaluation

A process for the testing, examination, and/or evaluation of a type under NTEP.

1.3. US/Canada Mutual Recognition Arrangement on Type Evaluation

A bilateral agreement reached by the United States and Canada which allows one country to recognize the examination of tests performed by the other country for certain devices. Both the United States and Canada operate type evaluation programs for weighing and measuring devices used in commercial applications. Each country will continue to issue its own (U.S.) Certificate of Conformance or (Canada) Notice of Approval, based on an evaluation completed by only one of the countries. *See Section 7.1 for the Agreement.*

2. Scope

Any submission for type approval of a device that consists of or contains metrologically significant software, either partially or wholly, should review these guidelines. This includes stand-alone software applications, software accompanying a submitted device, and software embedded in devices (firmware). In general, this document is relevant to any device that runs metrologically significant software and is being submitted for type approval.

This document includes requirements, considerations, and test procedures common to all software-based devices, including software-only products.

It is intended to be applied in conjunction with device-specific Pub. 14 documents.

3. Submission of Software for Type Evaluation

As part of the type evaluation submission, the manufacturer may be asked to provide the following to expedite the evaluation process of any metrologically significant software:

- The software identification (version, revision, checksum, etc.), and how to view it. See Section 5.
- An overview of the security aspects of the operating system, e.g. protection, user accounts, privileges, etc.
- A description of the software functions that are metrologically significant, meaning of the data, etc., e.g. an architecture diagram or flowchart.
- An overview of the system hardware, e.g. topology block diagram, type of computer(s), type of network, etc.
- For software-only products, a description of the minimum system requirements to run the software.
- A declaration of whether software separation is implemented and how it was accomplished. See Section 5.2.
- User manual(s), service manual(s), and/or other technical documentation.
- Dictionary of accessible commands (e.g., function keys or commands via external interfaces) available, accompanied by a description of the function of each command. This dictionary should be all-inclusive (i.e. no secret back door commands or functions).
- All necessary hardware, software, and accessories required to operate the software being evaluated.
- A representative example of each operating system, if multiple operating systems are supported.
- The means to achieve remote access, if it is required.

4. Markings

4.1. Certificate of Conformance number (CC) marking requirements

Built-For-Purpose: Reference Handbook 44 General Code S-1. Marking Requirements. There may also be specific code requirements in individual device-specific sections that must be complied with as well.

Not-Built-For-Purpose: Marking of the CC number becomes more complicated when dealing with software that runs on a general-purpose device such as an off the shelf PC or tablet/mobile phone. Hard marking the CC (the preferred method in most cases) is not preferable if the software is installed by the user or may run on devices other than what was submitted for type approval. In these cases, it is preferable to mark the CC continuously on the display. In cases where the CC mark is continuously displayed, it should not be possible to obstruct or overwrite this information when the device is operating.

If the CC can neither be hard-marked or continuously displayed, there will be allowed only a limited number of options to access the CC via the user interface (See Section 5.4).

4.2. Version / Revision Number Marking Requirements

Ideally, submitted software should continuously display the metrologically significant version number, similar to the CC. Hard marking is discouraged unless "absolutely necessary" (see wording in G-S.1.d.1.i & ii) with the understanding that software is more likely to be upgraded and the original hard marking would deviate from the actual version. Navigating to the version number via directions in the CC is permitted if the process is straightforward and easily understood.

See Handbook 44 General Code G-S.1. for additional marking requirements. Also see Section 5.4 of this document.

5. Software Identification

Marking requirements must comply with G-S.1. in Handbook 44, including requirements for version / revision. The following recommendations are intended as further guidance to satisfy the requirements regarding software identifiers.

5.1. Appropriate Means of Marking Metrologically Significant Software

5.1.1. Examples of Acceptable Software and Version / Revision Identifiers

Example 1: *Revision 1.XX.YY* – In this example, 1 is the metrologically significant version number, XX is a version number for non-metrologically significant software, and YY indicates bug fixes.

Example 2: *Ver. Number 1.XX.YYYYYYYY* – In this example, Ver. is an abbreviation for Version, 1 is the major revision, XX is the minor revision, and YYYYYYYY is the build date.

Example 3: *Ver.No. 1.XX YYYYYYYYYYYYYYYY* – In this example, Ver. is an abbreviation for Version, No.

is an abbreviation for Number, 1 is the major revision, XX is the minor revision, and YYYYYYYYYYYYYYYY is a hash of the metrologically significant executable code.

Example 4: *Version 1* – In this example, the version/revision consists of a single numeric value. There is no major/minor revision and no software separation is employed. The entire software is considered metrologically significant.

The manufacturer is not limited to these examples.

5.2. Software Separation and Marking Consequences

Manufacturers may choose to separate metrologically significant software from non-metrologically significant software. Separation would allow the revision of the non-metrological portion without the need for further evaluation. In addition, non-metrologically significant software may be updated on devices without breaking the seal, if so designed.

Separation of software requires that all software modules (programs, subroutines, objects, etc.) that perform metrologically significant functions or that contain metrologically significant data domains form the metrologically significant part of a measuring instrument (device or sub-assembly).

If the separation of the software is not possible or needed, then the software is metrologically significant as a whole. In that case, any modification to the software may result in NTEP requiring further review of the modifications to the software.

Where the version revision identifier is comprised of more than one part, the manufacturer shall describe which portion represents the metrological significant software and which does not.

5.2.1. Examples of Software Identifiers When Software Separation is Employed

Example 1: Version No. 1.XX – In this example, No. is an abbreviation for Number, 1 is the major revision version number, XX is the minor version number. Both pertain to metrologically significant software. There would typically be an entirely separate version identifier for the non-metrologically significant software.

Example 2: Rev. Number 1.XX – In this example, Rev. is an abbreviation for Revision, 1 is the version number for the metrologically significant software, and XX is the version number for the non-metrologically significant software.

5.3. Relationship Between Software and Software Identifier

The manufacturer should be able to describe and possibly demonstrate how the version or revision identifier is directly and inseparably linked to the metrologically significant software.

This means that the software can't be easily changed without changing the software identifier. For example, the version identifier can't be in a text file that's easily editable, or in a variable that the user can edit.

5.4. Presentation of Software Identifier

The software identifier shall be easily viewed by the field inspector. It can be constantly displayed or accessible via the display. Instructions for viewing the software identifier shall be described in the CC. The identifier must be accessible via a straightforward interaction of the interface, preferably requiring no navigation (directly from the display) or via navigation of menus or displays not more than two levels deep.

5.4.1. Example Icons and Menu Text

If the software identifier is accessible via the display, options to display it include a menu option, an icon like those in the list below, or some other method that was accepted during type approval.

Menu Text examples	Icon examples	Essential characteristics
Information Info		Top level menu text or icon • Icon text is a lower case "i" with block serifs • Text color may be light or dark but must contrast with the background color • Icon may have a circular border • Activation of this menu text/icon may invoke a second level menu text/icon that recalls metrology information.
Help ? About		Top level menu text or icon • Icon text is a question mark • Text color may be light or dark but must contrast with the background color • Icon may have a circular border • Activation of this menu text/icon may invoke a second level menu text/icon that recalls metrology information.
Metrology Metrological Information		Top or second level menu text or icon • Icon text is an upper case "M" • Text color may be light or dark but must contrast with the background color • Icon may have a circular, rectangular, or rounded rectangle border. • If present, the activation of this menu text/icon must recall at a minimum the NTEP CC number.
Weights & Measures Info	Boxes with W&M or W/M	Top level menu text or icon • W&M Info • Weights & Measures

5.4.2. Exceptions

Permanently marking the version or revision identifier shall be acceptable providing the device does not always have an integral interface to communicate the version or revision identifier.

An integral interface is one that is always present and might be a printer, remote console, display, etc.

6. Software Security

Software should be protected from unintentional or unauthorized changes. The manufacturer should be prepared to describe the methodology chosen.

Example: Integrity check, such as on each start-up the metrologically significant software calculates a checksum (or hash) of the program code and metrologically significant 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 metrologically significant software 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.

Facilitation of fraud, as defined in Handbook 44, General Code G-S.2.

Example: Confirmation that the change is intentional, such as the user is guided by menus. The metrologically significant 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.

7. Software Maintenance

Technical means shall be employed to guarantee the authenticity of the loaded software. Refer to facilitation of fraud, as defined in Handbook 44, General Code G-S.2.

When software is updated, the update itself can change one or more typical features or parameters to be sealed without these changes being reflected by a device's sealing method. For this reason, it is important that any update that changes the metrologically significant software be considered a sealable event as required by Handbook 44, General Code G-S.9 and in the section specific to the appropriate device type.

Example 1: Software updates are only able to be installed by authorized users, through a protected interface.

Example 2: Authentication of authorized software that is to be installed, such as signed update modules. 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 digital signature of the revised software or module using the private key in the manufactory. The public key is stored in a metrologically significant software module 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.

Example 3: There is no way to modify the metrologically significant software except via a switch protected menu. This switch is mechanically sealed in the inactive position, making modification of the metrologically significant software impossible. To modify the metrologically significant software, the switch needs to be activated, inevitably breaking the seal by doing so.

An event counter or event logger should be considered to be part of the protected software.

8. Metrologically Significant Parameters

Metrologically significant parameters have a direct influence on the compliance, performance, or operation of the software, as specified in Handbook 44, General Code G-S.8 and in the section specific to the appropriate device type.

Examples: Unit of measure, precision of measurement result, calibration factor

Further examples of metrologically significant parameters can be found in Publication 14 for the appropriate device type.

The manufacturer may be asked for a list of metrologically significant parameters.

Metrologically significant parameters should be protected from accidental or unintentional changes, or intentional misuse. Metrologically significant parameters need to be protected from changes by a sealing method. The manufacturer should be prepared to describe the methodology chosen.

Example: Metrologically significant parameters are only able to be changed by authorized users, through a protected interface.

9. Software Evaluation Checklist

When performing a type approval involving software, there is a checklist intended for the type evaluator to review with the manufacturer. See checklist. Many of the steps in the checklist are further clarified in this document.

10. NCWM Website Resources

NCWM website is: www.ncwm.com

Information may be printed or downloaded to individual personal computers. NTEP related information available includes:

- Active and Inactive Certificates of Conformance issued from January 1, 1986 to present

- NTEP Applications

- NTEP and Laboratory Fees

- List of NTEP Participating Laboratories

- NTEP Sectors

- International Recognition

- Conformity Assessment

- NTEP Logo

- Frequently Asked Question (FAQs)

Appendix A: Checklist for Devices with Software

1. Devices consisting of or contain metrological software

    1.1. Is there metrologically significant software in the system?   ☐ Yes ☐ No
        (If No, stop. This checklist does not apply.)

    1.2. Is the metrological software capable of being updated in the field? And,   ☐ Yes ☐ No ☐ N/A

    1.3. if yes, is the update of the metrological software protected by physical or electronic means, (i.e. is the ability to change the software protected by a seal?)   ☐ Yes ☐ No ☐ N/A

    1.4. The software documentation contains:

        1.4.1. a description of all functions, designating those that are considered metrologically significant.   ☐ Yes ☐ No ☐ N/A

        1.4.2. a description of the securing means (evidence of an intervention).   ☐ Yes ☐ No ☐ N/A

        1.4.3. a description of how to check the actual software identification.   ☐ Yes ☐ No ☐ N/A

    1.5. The software identification:

        1.5.1. includes: ☐ Yes ☐ No ☐ N/A
                1.5.1.1. version/revision
                1.5.1.2. model description
                1.5.1.3. manufacturer
                1.5.1.4. CC

        1.5.2. is clearly assigned to the metrologically significant software and functions.   ☐ Yes ☐ No ☐ N/A

        1.5.3. is provided by the device as documented.   ☐ Yes ☐ No ☐ N/A

        1.5.4. is directly linked to the software itself. This means that you can't easily change the software without changing the software identifier. For example, the version identifier can't be in a text file that's easily editable, or in a variable that the user can edit.   ☐ Yes ☐ No ☐ N/A

2. Programmable or Loadable Metrologically Significant Software

    2.1. The metrologically significant software is:

        2.1.1. Documented with all relevant information (see Section 3 for list of documents).   ☐ Yes ☐ No ☐ N/A

        2.1.2. Protected against accidental or intentional changes.   ☐ Yes ☐ No ☐ N/A

    2.2. Evidence of intervention (such as, changes, uploads, circumvention) is available until the next verification / inspection (e.g., physical seal, Checksum, Cyclical Redundancy Check (CRC), audit trail, etc. means of security).   ☐ Yes ☐ No ☐ N/A

3. Software with no access to the operating system and/or programs possible for the user. This section and section 4 are intended to be mutually exclusive.

    3.1. Check whether there is a complete set of commands (e.g., function keys or commands via external interfaces) supplied and accompanied by short descriptions.   ☐ Yes ☐ No ☐ N/A

4. Operating System and / or Program(s) Accessible for the User.

    4.1. Check whether a checksum or equivalent signature is generated over the machine code of the metrologically significant software (program module(s) subject to legal control Weights and Measures jurisdiction and type-specific parameters). This is a declaration or explanation by the manufacturer.   ☐ Yes ☐ No ☐ N/A

    4.2. Check whether the metrologically significant software will detect and act upon any unauthorized alteration of the metrologically significant software using simple software tools (e.g., text editor). This is a declaration or explanation by the manufacturer.   ☐ Yes ☐ No ☐ N/A

4.3. Check whether the manufacturer has provided a description of the software functions that are metrologically significant, meaning of the data, etc., e.g. an architecture diagram or flowchart. ☐ Yes ☐ No ☐ N/A

4.4. Check that there is guidance related to the software identification (version, revision, etc.), how to view it, and how it is tied to the software. ☐ Yes ☐ No ☐ N/A

4.5. Check that the manufacturer has provided an overview of the security aspects of the operating system, e.g. protection, user accounts, privileges, etc. ☐ Yes ☐ No ☐ N/A

5.   Software Interface(s)

5.1. Verify the manufacturer has documented:

5.5.1. that the program modules of the metrologically significant software are enumerated (if software separation is employed). ☐ Yes ☐ No ☐ N/A

5.5.2. the functions of the metrologically significant software that can be accessed. ☐ Yes ☐ No ☐ N/A

5.5.3. the metrologically significant parameters that may be exchanged via an external interface. ☐ Yes ☐ No ☐ N/A

5.5.4. that the description of the metrologically significant modules (functions) and data (parameters) is conclusive and complete. ☐ Yes ☐ No ☐ N/A

5.5.5. that if applicable, there are software interface instructions for the third party (external) application programmer. ☐ Yes ☐ No ☐ N/A