*XVII IMEKO World Congress*
*Metrology in the 3rd Millennium*
*June 22–27, 2003, Dubrovnik, Croatia*

# VALIDATION OF CUSTOM DEVELOPED SOFTWARE IN METROLOGY APPLICATIONS

*Tanasko Tasić, Roman Flegar*

Metrology Institute of the Republic of Slovenia (MIRS),
Laboratory for Information Technology in Metrology
Ljubljana, Slovenia

**Abstract** − Requirements for validation of custom developed software in metrology applications are mainly influenced by technical realisation of measuring instrument/system. In this article some issues related to the technical realisation of validated software are described. Furthermore, the state-of the art validation approaches and tools are presented, including failure risk analysis and test case selection. In the discussed example the validation approach in case of MIRS Mass Laboratory automation software is presented. Synthesis of presented methods, tools and techniques may be useful as guidance for validation of similar software applications.

**Keywords**: software validation, custom developed application, risk analysis, test case.

## 1. INTRODUCTION

The aim of this paper is to give an overview of acquired knowledge and experience in the validation of software in metrology applications gained in MIRS Laboratory for Information Technology in Metrology. Presented approaches may be combined and applied to validation of software in different fields of metrology.

When analysing a modern (automated) measuring system, it is obvious that it comprises, among other components of numerous software components. Those components are whether embedded in vendor supplied measuring instruments or custom ("do it yourself") laboratory developed, running on PC workstations. The reliability of measurement system as well as uncertainty of measurement results are in many cases dependent of the quality of software components used. Having that in mind it is inevitable to persuade both performer of measurements and supervision (i.e. accreditation) body of quality of automation software used.

Very often some commercial off-the shelf software (COTS) products are used for performance of particular data processing as well. Application of these software products has to be performed carefully, because they may contain certain deficiencies, which are very well addressed in [4].

Some tools for validation of metrological software are already available. For example, there are some public reference program libraries dedicated for software validation purposes available on Internet [6], [8]. They are suitable for validation of generic software functions (like circle, square, fitting algorithm …) that are building blocks of real metrological software applications.

In addition, there are some public databases with reference data for validation of particular metrological software applications also available in Internet. Those reference data are prepared for use in test cases for validation of certain software modules [5], [7].

For both reference software and reference data one must have in mind that they very rarely fulfil all needs of particular validation task and that they may be used for validation of particular building blocks only.

Another possibility for validation is intercomparison of metrological software modules. With this method modules of the same kind (performing calculation of same metrological function) may be validated in manner of classical laboratory intercomparison. For this approach specially designed (dedicated for every particular metrological function to be tested) reference data sets have to be produced and made available (i.e. on a server on the Internet) [9].

## 2. SOFTWARE VALIDATION

### *2.1 Basic principles*

When discussing software validation we can distinguish at least two situations when it is necessary to be performed. Those are:

- initial validation, when validation is done before first software use and
- later validation, which may or even, has to be done for various reasons (maintenance, virus detection, …) that are very well explained in [3].

The meaning and purpose of initial and later validations are very similar to initial and subsequent verifications of measuring instruments in legal metrology.

The validation of software in other words is software testing. There are two different understandings of software testing. One definition [1] says that software testing is reviewing and executing the program in order to detect faults (provoke failures). This approach stimulates removal of as many software faults as possible. That certainly is suitable from software manufacturer (developer) point of view. Another aspect is the aspect of end-user or inspection body: to verify equipment functionality. This approach is more practical. It does not require “complete” testing - only the functionality has to be examined, regardless of possible hidden “dormant” faults.

Although opposite at the first glance, those understandings have the same aim and lead to the common result – the software must be adequate for it’s purpose, that is to provide satisfactory equipment functionality in foreseen conditions of use.

### *2.2 Software quality requirements*

Another dilemma in approach to software testing is caused by the source of validation requirements. Different user groups are interested in different software quality characteristics, which can be grouped according to the field of instrument use.

In legal metrology applications requirements for the measuring instruments software are defined by legislation – national or international (in the manner of European directives, for example Measuring Instruments Directive – MID). Specifics of requirements for testing of software in legal metrology instruments compared to other applications is that more attention is focused on testing of non-desired functionality (what software should not do or allow) and software and data protection.

There is another category of devices where safety requirements are of great importance. Those are medical, transportation, devices used in power plants and military devices. For those applications requirements are defined in rigorous safety standards, whether for entire measuring system or for the particular software components.

Other requirements are caused by the market rules, like responsibilities and consequences caused by faulty product or lost of the market share in case of malfunction with severe consequences.

Requirements for custom developed software applications in metrological laboratories (other then legal metrology instruments) may be derived both from good laboratory praxis and from requirements of the international standard ISO/IEC 17025 (General requirements for the competence of testing and calibration laboratories).

## 3. PERFORMING OF VALIDATION

Analysis of software under test as well as test methods to be used have to be discussed and agreed between person performing validation and designer and user of measuring system.

### *3.1. Analysis of validated software*

The first step is to analyse of software to be tested. For this purpose appropriate documentation has to be supplied, including program description, and often program source code. Since complete testing of software is not possible, it is necessary to determine what program functions and modules have to be tested and under what conditions. Appropriate aid for this decision is so called “risk analysis” [2], during which focus for validation is defined. Entire software has to be analysed in manner presented on figure 1.

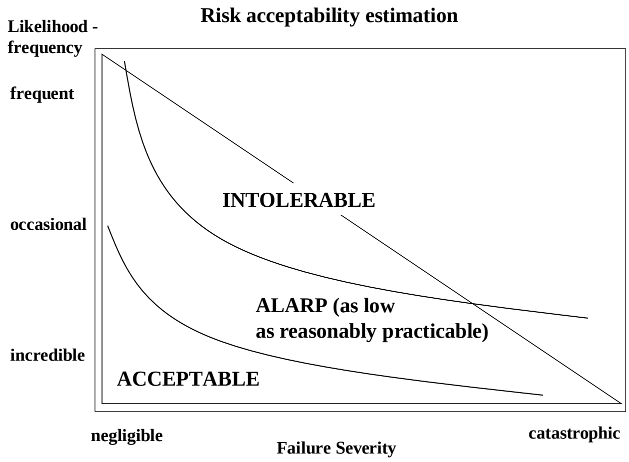


Figure 1: Risk acceptability estimation

Estimation has to be prepared for all components (modules) of validated software with likelihood of module outage (dependent also of frequency of module usage) and severity of possible failure, as shown on figure 1. Modules with combination likelihood/severity that lies in the “intolerable” area have to be tested in greater extent, with aim to find as much faults as possible, so their outage likelihood decreases and likelihood/severity falls into “acceptable” area.

Specific metrology applications are also distributed measurement systems where measurement data are transmitted via computer networks. Usually public networks like Internet are most convenient infrastructure for data transfer. There are several IT related aspects in such applications that have to be taken into account (also during the validation) - data security and integrity as well as the proper identification and authentication of the participants in data exchange. State of the art approach for solving these problems is application of asymmetric key infrastructure.

### *3.2. Test environment*

Testing of automation software is usually performed on dedicated workstations, for which the environment has to be preferable the same as in the place of use (i.e. hardware environment and operating system). Sometimes it is not possible in praxis to set up hardware environment entirely (i.e. it is not possible to bring to testing laboratory all measuring instruments that form the system that is operating in specific environmental conditions). In such cases some of measuring system components (in most cases functionality of complete measuring devices) or software modules has to be simulated. This is realised by software modules designed specially for that purpose and inserted in software under test to replace modules that can not be executed. Use of such modules is convenient but special attention has to be taken to avoid introducing new errors with this simulation software modules. Usually, such modules have to be validated separately, before inclusion in the validated automation software.

*3.3. Test cases selection*

Previous chapter describes distribution of attention paid to testing of particular modules according to the results of risk analysis. The next issue to be defined is selection of test cases.

Test case is set of pairs of values of input parameters and expected values of output from module under test [1]. There are several methods for selection of test cases. Probably the most appropriate for metrology applications is combination of method of equivalence partitioning and the method of boundary values. Method of equivalence partitioning implies existence of at least one test case for every algorithm branch of tested program. According to the method of boundary values, test cases must have values of input parameter very close (less than, equal and greater) to algorithm switching values (A and B on figure 2). Unexpected values (less than minimum and greater then maximum input parameter value) also have to be included in test sets.

Test cases selected in described manner are minimal for sufficient testing of one module. Those test cases may be used as reference for software validation in all situations.

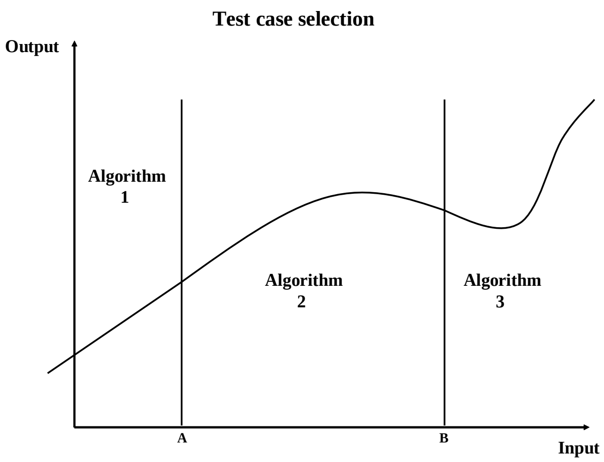


Figure 2: Selection of test cases

## 4. PRACTICAL EXAMPLE

As the practical example an approach to the validation of MIRS mass laboratory automation software will be presented. In the laboratory the calibration of weights is performed, according to comparison method. The software controls comparator balances and instruments for measuring of environmental conditions (temperature ***t,*** relative humidity ***h*** and atmospheric pressure ***p***). For practical reasons software modules for communication with those instruments were substituted with simulation subroutines which used previously recorded real raw data as input parameters.

Automation software is programmed in three programming languages – National Instruments LabVIEW® (communication modules and calculations), Microsoft Access® (database functions) and Microsoft Visual Basic® (graphical user interface).

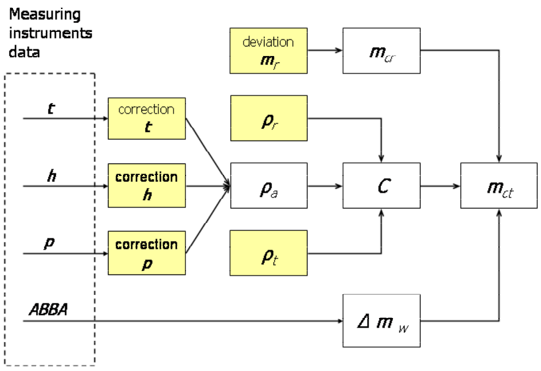


Figure 3: Example structure of software under test

Modules performing following calculations were selected during risk analysis as most important for thorough checking:

$$m_{ct} = m_{cr}(1 + C) + \Delta m_W \,, \tag{1}$$

where:
$m_{cr}$ - standard weight conventional mass,
$C$ - air buoyancy correction,
$\Delta m_W$ - mean value of the ABBA measured difference.

Air buoyancy correction is calculated as:

$$C = \frac{(\rho_r - \rho_t)(\rho_a - \rho_0)}{(\rho_r - \rho_0)(\rho_t - \rho_a)} \,, \tag{2}$$

where:
$\rho_r$ - density of standard weight,
$\rho_t$ - density of weight under test,
$\rho_a$ - air density,
$\rho_0$ - presumed air density 1,2 kgm$^{-3}$.

Weight density parameters are stored in database, air density is calculated according to:

$$\rho_a = \frac{0.34848p - 0.009024he^{0.0612t}}{237.15 + t} \tag{3}$$

where:
$p$ - atmospheric pressure,
$h$ - relative air humidity,
$t$ - temperature.

Test cases were selected according to the 3.2. The validation outcome showed good performance of the software under test in respect to desired functionality and expected range of input parameters. The recommendation of evaluators was that software should be improved in way to reject irregular values of input parameters [10].

## 5. CONCLUSIONS

Validation of custom designed software in metrology applications requires both knowledge of software testing as well as the physical background of the metrology field that software deals with.

Preparatory activities for testing of metrological software require thorough analysis of software design and applied technology in order to focus on testing of software modules that are most critical for performance of software under test and select proper test methods. Although several tools are available, including reference software and reference datasets, sufficient testing of metrological software is not a routine task. Those tools are usually not sufficient or not directly applicable and tester has to modify them or develop its own tools, - reference software, datasets and modules for simulation of particular functionality of system under test.

Although international standard ISO/IEC 17025 says that commercial off-the shelf software products may be considered to be sufficiently validated; it is sometimes good not to believe that entirely and to validate functions performed by such software as well.

Generally speaking, it is important to realise that tolls that will solve all validation problem do not exist. Sufficient validation requires knowledge about physical phenomena automated with the software under test, thorough analysis of the software and application of available tools, methods and software testing techniques.

## REFERENCES

[1] Myers J. G.: The Art of Software Testing; *John Wiley & Sons, Inc.*, 1979.

[2] Debra S. Herrmann, “Software Safety and Reliability”, *IEEE Computer Society*, 1999.

[3] EA/GA(98)95: EA Guidelines for the Use of Computers and Computer Systems in Accredited Laboratories, 1995.

[4] SSfM (Software Support for Metrology): Testing Spreadsheets and Other Packages Used in Metrology, various reports, *Internet pages of the program Software Support for Metrology:* “http://www.npl.co.uk/ssfm”.

[5] NIST (National Institute for Standards and Technology): Statistical Reference Datasets: *Internet pages “http://www.nist.gov/srd/”*.

[6] MetroS Key Functions, *program MetroS (Metrological Software) internet pages*: “http://www.npl.co.uk/ssfm/metros/key_functions”

[7] MetroS Reference Data Sets, *program MetroS (Metrological Software) internet pages*: ”http://www.npl.co.uk/ssfm/metros/reference_data_sets/”

[8] NIST (National Institute for Standards and Technology): *Guide to Available Mathematical Software reference libraries,* internet pages: http://gams.nist.gov/Problem.html

[9] *Tasić*, T., *Bojkovski,* J.*:* Intercomparison of metrological software modules-can it be useful for metrological laboratories? Joint BIPM-NPL Workshop on the Impact of Information Technology in Metrology, National Physical Laboratory, Teddington, UK, 16-19 September 2002.

[10] Tasić, T., Oblak, M.. Validation of the automation software in the MIRS Mass Laboratory, 11th International Electrotechnical and Computer Science Conference ERK 2002, 23. - 25. September 2002, Portorož, Slovenia.

**Authors:**
Tanasko Tasić and Roman Flegar;
Metrology Institute of the Republic Of Slovenia, Laboratory Centre Ljubljana,
Laboratory for Information Technology in Metrology, Grudnovo nabrežje 17, 1000 Ljubljana, Slovenia, tel: +386 1 24 42 700, fax: +386 1 24 42 714, e-mail: tanasko.tasic@gov.si; roman.flegar@gov.si