NEW06 TraCIM

Traceability for computationally-intensive metrology: Specification of computational aims

Deliverable 2.1.3–A v1.0
of Work Package
WP2 (Formal statement of computational aim)

Authors:

Alistair Forbes, Peter Harris, Keith Lines
National Physical Laboratory, UK

A report of the EMRP joint research project
NEW06 "Traceability for computationally-intensive metrology"

1. Work package	**2. Deliverable number**	**3. Reporting date**
WP2	D2.1.3–A	November 2013

4. Title
Specification of computational aims

5. Author(s)	**6. Lead author (e-mail)**
Alistair Forbes, Peter Harris, Keith Lines	peter.harris@npl.co.uk

7. Contributing researchers (institutes)
Gertjan Kok (VSL), Ian Smith and Minh Hoang (NPL), Ulrich Lunze (WHZ)

8. Other contributing work packages	**9. Researchers in other WPs**
WP1	

10. Supplementary notes
This report is an internal project report.

11. Abstract
It is only possible to verify and validate software when it is known what problem the software is intended to solve or task the software is intended to execute. A statement of the computational aim of the software is used to set the user and functional requirements for the software developer, that is, to specify what is required of software to be a conforming product, and to provide a basis for the verification and validation of a software implementation. The aim of this report is to describe a generic approach to specifying a computational aim and to illustrate that approach with some examples. A companion report describes the mathematical and ICT (information and communications technology) tools used to capture the specification of a computational aim.

12. Key words
Traceability, computational aim, specification.

Contents

1 Introduction **1**

2 Requirements of a computational aim **1**

3 Specifying the computational aim of the mathematical problem **2**
3.1 Unique identifier . . . 3
3.2 Language . . . 3
3.3 Title . . . 3
3.4 Keywords . . . 4
3.5 Mathematical area . . . 4
3.6 Dependencies . . . 5
3.7 Input parameters . . . 5
3.8 Output parameters . . . 7
3.9 Mathematical model . . . 7
3.10 Signature . . . 7
3.11 Properties . . . 7
3.12 References . . . 7
3.13 Additional notes . . . 7
3.14 History . . . 8

4 Specifying a refinement of a computational aim **8**
4.1 Unique identifier . . . 10
4.2 Subtitle . . . 10
4.3 Metrology area . . . 10
4.4 Dimension or reference . . . 11
4.5 Values . . . 12
4.6 Additional keywords, dependencies, properties, references and notes . . . 12
4.7 History of refinement . . . 12

5 Example 1 **12**

6 Example 2 **15**

7 Example 3 **17**

8 Summary and conclusions **21**

Acknowledgements **21**

References **21**

1 Introduction

Metrological traceability is the property of a measurement result whereby the result can be related to a reference, such as a primary standard or unit, through a documented, unbroken chain of calibrations, each contributing to the measurement uncertainty [1]. When the traceability chain involves computation, it is necessary that the computational links in the chain are recognised explicitly and treated as any other link. For example, computational links should be shown to be operating correctly and, when they involve approximations, consideration given to their contribution to the measurement uncertainty. Furthermore, when software is used to deliver a measurement result, metrological traceability requires the documentation of all aspects in the process of developing, implementing and testing the software so that each stage in the process can be understood, checked and reproduced.

It is only possible to verify and validate software when it is known what problem the software is intended to solve or task the software is intended to execute. A statement of the computational aim of the software is used to set the user and functional requirements for the software developer, that is, to specify what is required of software to be a conforming product, and to provide a basis for the verification and validation of a software implementation. The aim of this report is to describe a generic approach to specifying a computational aim and to illustrate that approach with three examples: one that is typical of requirements arising in dimensional metrology, and two that are interdisciplinary being concerned with measurement uncertainty evaluation and key comparison data evaluation. The audience of the report is the members of the TraCIM network. Although others (outside the TraCIM network) may propose a specification of a computational aim, it is anticipated that those involved in the TraCIM system will review and release that specification. A companion report [2] describes the mathematical and ICT (information and communications technology) tools used to capture the specification of a computational aim.

The report is organised as follows. Section 2 discusses the requirements of a (specification of a) computational aim, and sections 3 and 4 describe the information, organised into a set of 'fields', proposed to specify a computational aim. Sections 5–7 give some examples of computational aims specified in this way. A summary and conclusions are given in section 8.

For a glossary of terms, see [3].

2 Requirements of a computational aim

A computational aim is required to state *what* problem the software is intended to solve or task the software is intended to execute and not *how* the problem is to be solved or task is to be executed [4]. Decisions about how the task is to be executed, such as the choice of algorithm, are the concern of the software developer whose responsibility it is to implement the computational aim, and can depend, for example, on the environment in which the algorithm is to be implemented and used.

The specification of the computational aim should be unambiguous, complete, free from contradictions, and independent of the environment, such as hardware and software configurations, in which it is to be implemented. The task 'calculate the mean of a set of numbers' is not complete: how many numbers are to be input? is the arithmetic or geometric mean to be calculated? The task 'perform task A when switch S is in position 1 or 2 and task B when switch S is in position 2 or 3' includes a contradiction: what task is to be performed when the switch is in position 2? Neither statements provide acceptable (specifications of) computational aims.

The use of natural language to specify a computational aim can lead to specifications that are verbose and ambiguous. (Indeed, ambiguity is often exploited in the everyday use of natural language.) A difficulty of using a programming language to specify a computational aim is that the behaviour of the programme can depend on, for example, the choice of hardware and compiler options. In addition, translation between programming languages can be problematic, particularly when language-specific features are used. Instead, it is proposed to use the abstract and universal language of *mathematics* to provide specifications of computational aims, which include propositional and predicate calculus [5] and formal description techniques and languages used in formal methods [4]. (In fact, only constructive mathematics is considered to avoid possible difficulties associated with incompleteness theorems, etc.)

Furthermore, it is proposed to distinguish as far as possible between the purely *mathematical problem* that is to be solved, which involves operations on numerical values, and an instantiation of the mathematical problem within a metrology area, measuring system or instrument, for which the numerical values will be associated with quantities with given dimensions, measurement units and (possibly) sets of typical values. In this way, the task 'calculate the arithmetic mean of ten lengths with the units of the metre' is considered as a *refinement* of the task 'calculate the arithmetic mean of ten lengths' that, in turn, is a refinement of the mathematical task 'calculate the arithmetic mean of ten real numbers'. (In the context of formal methods, 'refinement' is used to describe a process of producing more detailed specifications of a computational aim that are closer to a software implementation. Here, we use the term to mean a process of producing more detailed specifications that are closer to the application of the computational aim within a particular metrology area.) Whereas general reference problems (defined by a reference data set and corresponding reference results) are associated with the computational aim for a mathematical problem, customised or bespoke reference problems are associated with a refinement of such a computational aim.

3 Specifying the computational aim of the mathematical problem

The specification of the computational aim of an underpinning mathematical problem is composed of information contained in the following fields:

1. Unique identifier;
2. Language;
3. Title;
4. Keywords;
5. Mathematical area;
6. Dependencies;
7. Input parameters:
   (a) Symbol;
   (b) Description;
   (c) Type;
   (d) Shape;
   (e) Constraints;
8. Output parameters:
   (a) Symbol;
   (b) Description;

   (c) Type;
   (d) Shape;
   (e) Constraints;

9. Mathematical model;
10. Signature;
11. Properties;
12. References;
13. Additional notes;
14. History.

The fields are described below. Those fields that are mandatory and searchable are indicated. A specification of a computational aim will not be considered complete unless information is provided for all mandatory fields. A field is searchable if the set of all computational aims for which the field (or a part of the field) takes a prescribed value can be identified, i.e., the set is unique and can be constructed in a finite time using a finite number of operations. The value may be prescribed from a defined set of values (as in a drop-down list) or as free-text.

3.1 Unique identifier

A unique identifier assigned by the TraCIM system using information in subsequent fields that takes the form

<language>/<mathematical area>/<refinement number>/######

where '######' is a six digit positive integer containing leading zeros as necessary, which is incremented by the TraCIM system each time a computational aim is approved. The field is mandatory and searchable, and should be human-readable. If a mathematical area is not specified, the 'dash' symbol (-) is used in its place. The refinement number for the specification of the computational aim for an underpinning mathematical problem is zero. Examples of valid identifiers would be

en/I1a1a/0/000001

and

de/-/0/000099

(see sections 3.2 and 3.5).

3.2 Language

The natural language in which the computational aim is written, expressed as the full language name and its abbreviation [6] . The field is mandatory and searchable. Examples include English (en), French (fr) and German (de).

3.3 Title

A title or short statement of the computational aim. The field is mandatory and searchable.

3.4 Keywords

A list of keywords. The field is not mandatory but is searchable.

3.5 Mathematical area

The mathematical area to which the computational aim belongs. The field is not mandatory but is searchable. A classification based on the 'Guide to Available Mathematical Software' (GAMS) index [7] is used. The highest level of the index contains the following broad subject areas:

A Arithmetic, error analysis;

B Number theory;

C Elementary and special functions;

D Linear algebra;

E Interpolation;

F Solution of nonlinear equations;

G Optimization;

H Differentiation, integration;

I Differential and integral equations;

J Integral transforms;

K Approximation;

L Statistics, probability;

M Simulation, stochastic modeling;

N Data handling;

O Symbolic computation;

P Computational geometry;

Q Graphics;

R Service routines;

S Software development tools;

Z Other.

These subject areas are further subdivided into more specific problem areas. For example, the area

I Differential and integral equations

is subdivided into the classes

I1 Ordinary differential equations (ODE's),

I2 Partial differential equations, and

I3 Integral equations (**I3**),

each of which are themselves further subdivided. For example, the class **I1a1a** is constructed as

I Differential and integral equations,

I1 Ordinary differential equations (ODE's),

I1a Initial value problems,

I1a1 General, nonstiff or mildly stiff,

I1a1a One-step methods (e.g., Runge-Kutta),

and is not further subdivided.

3.6 Dependencies

A list of dependencies on other computional aims (with their unique identifiers). The field is not mandatory.

3.7 Input parameters

The parameters that must be assigned in order for the computational aim to be executed. The field is mandatory. The number of input parameters is specified and then for each input parameter the following mandatory information is provided:

1. Symbol;
2. Description;
3. Type, such as
   (a) unsigned integer $\mathcal{N}_0$, with values in the set $\{0, 1, 2, \ldots\}$,
   (b) signed integer $\mathcal{Z}$, with values in the set $\{\ldots, -2, -1, 0, 1, 2, \ldots\}$,
   (c) real $\mathcal{R}$,
   (d) complex $\mathcal{C}$;
   (e) boolean $\mathcal{B}$, with values 'true' or 'false', or 'T' or 'F', or '0' or '1', etc.,
   (f) character or string $\mathcal{S}$, or
   (g) mathematical function $\mathcal{F}$;
4. Shape;
5. Constraints or function signature.

For parameters of type other than $\mathcal{S}$ and $\mathcal{F}$, the shape of the parameter can be scalar, (row or column) vector of length n, matrix of dimension $n_1 \times n_2$, or k-dimensional matrix of dimension $n_1 \times n_2 \times \cdots \times n_k$. For parameters of type $\mathcal{S}$, the shape of the parameter can be single character, string of length n, (row or column) vector of length m of single characters, or (row or column) vector of length m of strings of length n. For parameters of type $\mathcal{F}$, the shape of the parameter shall be the same as for the output parameter of the function.

For parameters of type other than $\mathcal{F}$, a constraint can be expressed as a mathematical condition. For example, the constraint (or property) that a real scalar parameter takes a value x in the finite interval (a, b) may be expressed as $a < x < b$ or $x \in (a, b)$. A general approach to specifying a constraint on the parameter is to use a *predicate*, i.e., a boolean-valued function of the parameter that is true when the value of the parameter satisfies the constraint and false otherwise. For example:

- The constraint (above) that a real scalar parameter takes a value x in the finite interval (a, b) may be expressed as
$$\{x : P(x)\}, \quad P(x) = ((a < x) \wedge (x < b)),$$
where $a < x$ and $x > b$ are propositions each of which is either true or false, $\wedge$ denotes the logical operator 'AND', $=$ denotes assignment, and $P(x)$ is the predicate.

- A real scalar parameter with the property that it is a *constant* can be defined by the constraint that the parameter takes a single value x equal to the constant, i.e.,
$$\{x : P(x)\}, \quad P(x) = (x == c),$$
where $==$ denotes the relational operator 'EQUALS', and c is the constant.

- A (row) vector boolean parameter with the property that at least one component of the parameter must take the value 'true' is expressed as
$$\{\boldsymbol{b} : P(\boldsymbol{b})\}, \quad P(\boldsymbol{b}) = (b_1 \vee \cdots \vee b_n),$$
where $\boldsymbol{b} = (b_1, \ldots, b_n)$ denotes a value of the parameter, and $\vee$ denotes the logical operator 'OR'.

- The constraint that a character string parameter takes a value s from a specified set of possible strings, e.g., {'Y', 'y', 'N', 'n'} is expressed as
$$\{s : P(s)\}, \quad P(s) = ((s == \text{'Y'}) \vee (s == \text{'y'}) \vee (s == \text{'N'}) \vee (s == \text{'n'})).$$

A parameter of type $\mathcal{F}$ is used to express a mathematical relationship f between a set of (dummy) parameters, denoted by $X_1, \ldots, X_p$, and a single (dummy) parameter, denoted by Y, in the generic form
$$Y = f(X_1, \ldots, X_p).$$
A 'constraint' on the function is used to specify the signature of the function, which identifies the input and output parameters of the function, and constraints on the input and output parameters are used to specify, respectively, the domain and co-domain of the function. For example:
$$\{f : Y = f(X_1, \ldots, X_p), X_1 \in \mathcal{R}, X_2 \in \mathcal{R}, \ldots, X_p \in \mathcal{R}, Y \in \mathcal{R}\},$$
or
$$\{f : (X_1 \in \mathcal{R}, \ldots, X_p \in \mathcal{R}) \mapsto Y \in \mathcal{R}\}.$$

3.8 Output parameters

The parameters that are assigned as a result of executing the computational aim. The field is mandatory. An output parameter can be of type unsigned integer $\mathcal{N}_0$, signed integer $\mathcal{Z}$, real $\mathcal{R}$, complex $\mathcal{C}$, boolean $\mathcal{B}$ or character or string $\mathcal{S}$. The number of output parameters is specified and then each output parameter is described by the same information as for an input parameter (section 3.7). A parameter can be included as both an input and output parameter if some aspect of the parameter, such as its value or the constraints on its value, is modified by the execution of the computational aim.

3.9 Mathematical model

A statement of what problem is intended to be solved or task is intended to be undertaken. The field is mandatory. The mathematical model provides a description of the relationships between the input and output parameters of the computational aim. It can be expressed using mathematical notation, in a formal specification language or as pseudo-code that is independent of any intended implementation of the computational aim. The description of the mathematical model might not be unique, but it should be unambiguous, complete and free from contradictions.

3.10 Signature

A statement (in the form of a function signature) that is indicative of how software implementing the computational aim would be called. The field is mandatory. The signature is expressed using a dummy name for the function, but would clarify the parameters listed above that are input parameters, output parameters and input/output parameters.

3.11 Properties

A list of properties of the computational aim, such as uniqueness of a solution, a mathematical characterisation of the solution, information about the sensitivity of the solution to perturbations in the values of the input parameters, and conditions (constraints) that apply to combinations or functions of the input and output parameters. Such properties can be useful as the basis for generating reference data and corresponding reference results to test a software implementation of the computational aim. The field is not mandatory.

3.12 References

A list of references to supporting papers, reports, guides and documentary standards (with internet links where they exist). The references should relate to open or publically-available documents and not documents that are restricted or commercially-sensitive. The field is not mandatory.

3.13 Additional notes

Any notes that might help with understanding and implementing the computational aim. The field is not mandatory. For example, notes may include:

- Information pertaining to an implementation of the computational aim;
- A reference to a national or international database, such as the BIPM's Key Comparison Database (KCDB), that provides a metrological context to the computational aim;
- A list of typical metrological applications. For example, for a Chebyshev element calculation:

1. Application 1: metrology area length, branch coordinate metrology, inspection of form deviation;
2. Application 2: metrology area electromagnetic, ...

3.14 History

A history of the computational aim. The field is mandatory. The following information is provided:

1. Date created;
2. Author;
3. For each amendment to the computational aim:
   (a) Date of amendment;
   (b) Author of amendment;
   (c) Summary of amendment.

4 Specifying a refinement of a computational aim

The computational aims considered in section 3 are intended to be generic and not to relate to a particular metrology area, measuring system or instrument. A 'refinement' of the specification of such a computational aim gives *context* to the computational aim in terms of a metrology area, measuring system or instrument. The refinement involves providing information in additional, optional fields (identified below in red italics). All other fields shall remain unchanged, except for the unique identifier, which is also written in red italics.

A number of refinements of a computational aim is possible. A first refinement may associate dimensions (and units) with the quantities involved in specifying the computational aim. A second refinement may associate a particular sets of values with those quantities. An original (parent) specification need not exist before a refinement is developed. For example, the interest may only be in the specification of a computational aim related to a particular metrology area. However, it may be useful leter to develop the (more generic) parent specification in order to collect together refinements of it that are specific to different metrology areas. If a parent specification exists, each refinement should be traceable to it, and should inherit any changes made to it. Establishing treaceability of a software implementation to a refinement of a computational aim may be weaker than establishing traceability to the parent computational aim because the traceability only applies within the context of the refinement, e.g., to problems defined by typical values of the input parameters to the computational aim.

1. *Unique identifier;*
2. Language;
3. Title;
   (a) *Subtitle;*
4. Keywords;
   (a) *Additional keywords;*
5. Mathematical area;

   (a) *Metrology area;*

6. Dependencies;
   (a) *Additional dependencies;*
7. Input parameters:
   (a) Symbol;
   (b) Description;
   (c) Type;
   (d) Shape;
   (e) Constraints;
   (f) *Dimension or reference;*
   (g) *Values;*
8. Output parameters:
   (a) Symbol;
   (b) Description;
   (c) Type;
   (d) Shape;
   (e) Constraints;
   (f) *Dimension or reference;*
   (g) *Values;*
9. Mathematical model;
10. Signature;
11. Properties;
    (a) *Additional properties;*
12. References;
    (a) *Additional references;*
13. Additional notes;
    (a) *Further additional notes;*
14. History;
    (a) *History of refinement.*

4.1 Unique identifier

A unique identifier assigned by the TraCIM system using information in subsequent fields that takes the form

<language>/<mathematical area>/<metrology area>/<refinement number>/######

where <refinement number> is a positive integer. Examples of valid identifiers would be

en/I1a1a/L/1/000001

that identifies the first refinement of the computational aim with identifier ‘en/I1a1a/0/000001’, and

de/-/INT/2/000099

that identifies the second refinement of ‘de/-/0/000099’.

4.2 Subtitle

A subtitle to the title of the computational aim giving context to the refinement of the computational aim. The field is mandatory.

4.3 Metrology area

The metrology area to which the computational aim belongs. The field is mandatory and searchable. A classification based on the Consultative Committees of the International Committee for Weights and Measures (CIPM) is used [8]:

EM Electricity and magnetism;

PR Photometry and radiometry;

T Thermometry;

L Length;

TF Time and frequency;

RI Ionizing radiation;

U Units;

M Mass and related quantities;

QM Amount of substance–metrology in chemistry;

AUV Acoustics, ultrasound and vibration.

The further metrology area, viz.,

INT Interdisciplinary

is also included, which may be considered appropriate for generic calculations that are important to a number of the specific areas listed above.

4.4 Dimension or reference

The *dimension* of an input or output parameter expresses the dependence of the parameter on base quantities in a system of quantities. The field is mandatory. The dependence takes the form of a product of powers of factors corresponding to those base quantities omitting any numerical factor [1, clause 1.7]. For this purpose it is recommended that the International System of Quantities (ISQ) is used [1, clause 1.6], which is founded on the seven base quantities of

- length (L),
- mass (M),
- time (T),
- electric current (I),
- thermodynamic temperature (Θ),
- amount of substance (N), and
- luminous intensity (J).

For example, in the ISQ, the dimension of force is MLT^{-2}. The dimension of the parameter is given when it is not necessary to be specific about the reference (see below) of the parameter. For example, a computational aim may be valid when the measurement unit of a parameter is given as $\mathrm{kg\,m\,s^{-2}}$ or as $\mathrm{g\,cm\,s^{-2}}$. However, it can be expected that parameters having the same dimension will also have the same reference.

The *reference* of the parameter can be a measurement unit, measurement procedure or reference material. When a *measurement unit* [1, clause 1.9] is specified, it is recommended that the International System of Units (SI) [1, clause 1.16] is used for this purpose. The SI is founded on the measurement units for the base quantities of the ISQ, namely

- the metre (m) for length,
- the kilogram (kg) for mass,
- the second (s) for time,
- the ampere (A) for electric current,
- the kelvin (K) for thermodynamic temperature,
- the mole (mol) for amount of substance, and
- the candela (cd) for luminous intensity,

together with units derived from the base units, and multiples and sub-multiples of those units. For example, in the SI, the measurement unit of force might be given as $\mathrm{kg\,m\,s^{-2}}$, when expressed in terms of the base units, or as N (newton), when expressed in terms of a derived unit, or as $\mathrm{g\,cm\,s^{-2}}$, when expressed in terms of sub-multiples of the base units. The measurement unit for a parameter of dimension one (or dimensionless parameter) can be given as ‘one’ or ‘1’. In some cases the measurement unit for a dimensionless parameter is given a special name, e.g., radian, steradian, decibel or international normalised ratio, or is expressed by a quotient, e.g., millimole per mole equal to 10^3 or microgram per kilogram equal to 10^9.

An example of a parameter for which the reference is a *measurement procedure* is the 'Rockwell C hardness of a given sample', whose value might be given as '43.5 HRC' where 'HRC' denotes a measurement procedure to determine the Rockwell-C hardness of a sample block of material [1, clause 1.19].

An example of a parameter for which the reference is a *reference material* is the 'arbitrary amount-of-substance concentration of lutropin in a given sample of human blood plasma (WHO International Standard 80/552 used as a calibrator)', whose value might be given as '5.0 IU/l', where 'IU' stands for 'WHO International Unit' [1, clause 1.19].

4.5 Values

Typical values for an input or output parameter may be expressed as a single (fixed) value, an interval (or set of intervals), or as a state-of-knowledge probability distribution. The latter may be used when there is knowledge about the values of a parameter in the form of an estimate and associated uncertainty. The field is not mandatory.

4.6 Additional keywords, dependencies, properties, references and notes

Aditional information giving context to the refinement of the computational aim. The fields are not mandatory.

4.7 History of refinement

A history of the refinement of the computational aim. The field is mandatory.

5 Example 1

This example is concerned with specifying the computational aim for a task, viz., determining a best-fit geometric element from co-ordinate data obtained using a co-ordinate measuring machine (CMM), that is typical of requirements arising in dimensional metrology and important for making decisions about manufactured parts. Comparable specifications will apply for computational aims involving other geometric elements and different best-fit criteria, such as orthogonal distance regression and Chebyshev (minimum zone) regression. The computational aim is specified in two parts. The first part considers the underpinning mathematical problem of determining a best-fit (i.e., minimum circumscribed) geometric element (i.e., circle) to co-ordinate data restricted to the xy-plane. The second part considers a refinement of the specification given in the first part to the case that the co-ordinates of the data have the dimension of length and units of the metre, and are obtained using a CMM with a working volume of $1\ \mathrm{m} \times 1\ \mathrm{m} \times 1\ \mathrm{m}$ measuring in a plane orthogonal to the z-axis of the CMM.

In this example the mathematical model takes the form of an optimisation problem expressed in terms of the parameters of the geometric element to be determined and the provided measured data (see below). The example illustrates a computational aim for which the input and output parameters are numeric, but take different shapes (scalar, vector and matrix) and different numerical types (integer and real). Furthermore, it is possible to state properties of the solution to the optimisation problem in the form of a mathematical characterisation of that solution that can be used as the basis for generating reference data and corresponding reference results.

A natural language description of the optimisation problem is 'determine the circle of smallest radius that circumscribes the given measured co-ordinate data'. Introducing $(x_i, y_i),\ i = 1, \ldots, m$, to denote the co-ordinate data, (x_0, y_0) the co-ordinates of the circle centre, and r the circle radius, the optimisation problem can be expressed mathematically as: determine values (x_0^*, y_0^*, r^*) of (x_0, y_0, r) to solve

$$\min_{x_0, y_0, r} r \quad \text{such that} \quad (x_i - x_0)^2 + (y_i - y_0)^2 - r^2 \le 0,\ i = 1, \ldots, m.$$

An alternative expression of the computational aim is: determine values x_0^*, y_0^*, r^* to satisfy

$$\{(x_0^*, y_0^*, r^*) \in C\} \wedge \{\forall\, (x_0, y_0, r) \in C, r^* \le r\},$$

where

$$\{(x_0, y_0, r) \in C\} \Leftrightarrow \{\forall i, (x_i - x_0)^2 + (y_i - y_0)^2 - r^2 \le 0\}.$$

Here, C is the set of all circles (x_0, y_0, r), defined by centre co-ordinates (x_0, y_0) and radius r, that circumscribe the given measured co-ordinate data, and the circle defined by (x_0^*, y_0^*, r^*) is chosen as that element of C having smallest radius. Both constitute valid descriptions of the mathematical model for the task.

The specification of the underpinning mathematical problem is:

Unique identifier	en/-/0/000001
Language	English (en)
Title	determine minimum circumscribed circle to data in the xy–plane
Keywords	3
1	geometric element
2	circle
3	minimum circumscribed
Mathematical area	-
Dependencies	None
Input parameters	2
1	m
	number of data points
	$\mathcal{N}_0$
	scalar
	$m > 0$
2	$\boldsymbol{X}$, with $\boldsymbol{X}_{i,1} = x_i, \boldsymbol{X}_{i,2} = y_i$
	coordinates of data points
	$\mathcal{R}$
	matrix of dimension $m \times 2$
	none
Output parameters	2
1	$\boldsymbol{X}_0 = (x_0, y_0)$

	coordinates of circle centre
	$\mathcal{R}$
	vector of length 2
	none
2	r
	radius of circle
	$\mathcal{R}$
	scalar
	$r \geq 0$
Mathematical model	given $(x_i, y_i),\ i = 1, \ldots, m$, determine values (x_0^*, y_0^*, r^*) of (x_0, y_0, r) to solve $$\min_{x_0, y_0, r} r \quad \text{such that} \quad (x_i - x_0)^2 + (y_i - x_0)^2 - r^2 \leq 0,\ i = 1, \ldots, m$$
Signature	$[\boldsymbol{X}_0, r] = \text{MCCircle2d}(m, \boldsymbol{X})$
Properties	4
1	for $m = 1$, the data point defines a circle of zero radius
2	for $m = 2$, the data points define a diameter of the solution circle
3	for $m > 2$, the solution circle interpolates a subset of the data points for which either (a) two of the interpolated points define a diameter of the circle, or (b) three of the interpolated points define an acute-angled triangle
4	a single global solution exists
References	1
1	G T Anthony, H M Anthony, B Bittner, B P Butler, M G Cox, R Drieschner, R Elligsen, A B Forbes, H Gross, S A Hannaby, P M Harris and J Kok, Reference software for finding Chebyshev best-fit geometric elements, *Precision Engineering*, **19**, 28–36, 1996
Additional notes	None
History	created 2012-11-05 by Peter Harris (NPL, UK)

The additional information necessary for the specification of the refinement of the computational aim is:

Unique identifier	en/-/L/1/000001
Subtitle	measurements made by co-ordinate measuring machine
Additional keywords	2

1	dimensional metrology
2	co-ordinate measuring machine
Metrology area	L
Additional dependencies	en/-/0/000001
Input parameters	2
1	dimensionless
	$10 \leq m \leq 1\,000$
2	dimension L, unit m
	$-0.5 \leq x_i \leq 0.5,\ -0.5 \leq y_i \leq 0.5$
Output parameters	2
1	dimension L, unit m
	$-0.5 \leq x_0 \leq 0.5,\ -0.5 \leq y_0 \leq 0.5$
2	dimension L, unit m
	$0 \leq r \leq 1$
Additional properties	None
Additional references	None
Additional notes	None
History of refinement	created 2012-11-05 by Peter Harris (NPL, UK)

6 Example 2

This example is concerned with specifying the computational aim for a task, viz., evaluating an estimate and associated standard uncertainty for a measured quantity, that is interdisciplinary. It illustrates a computational aim for which there are input parameters that take the form of mathematical functions. The mathematical model for the computational aim is expressed as a set of formulæ that relate the output parameters to the input parameters.

Unique identifier	en/-/0/000002
Language	English (en)
Title	evaluate estimate and associated standard uncertainty by propagation of distributions
Keywords	3
1	estimate
2	standard uncertainty

3	propagation of distributions
Mathematical area	-
Dependencies	None
Input parameters	3
1	N
	number of input quantities
	$\mathcal{N}_0$
	scalar
	$\{N : P(N)\}, \quad P(N) = (N > 0)$
2	f
	measurement model, which maps values $\boldsymbol{\xi}$ of the input quantities to values η of the output quantity
	$\mathcal{F}$
	scalar
	$\{f : \eta = f(\boldsymbol{\xi}), \boldsymbol{\xi} \in \mathcal{R}^N, \eta \in R\}$
3	g
	joint probability density functon for input quantities, which maps values of the input quantities to values p of probability density
	$\mathcal{F}$
	scalar
	$\{g : p = g(\boldsymbol{\xi}), \boldsymbol{\xi} \in \mathcal{R}^N, p \in \mathcal{R}, p \geq 0, \int_{-\infty}^{\infty} p \, \mathrm{d}\boldsymbol{\xi} = \int_{-\infty}^{\infty} g(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\xi} = 1\}$
Output parameters	2
1	y
	estimate of output quantity
	$\mathcal{R}$
	scalar
	none
2	u_y
	standard uncertainty associated with y
	$\mathcal{R}$
	scalar

	$\{u_y : P(u_y)\}, \quad P(u_y) = (u_y \geq 0)$
Mathematical model	evaluate $$y = \int_{-\infty}^{\infty} \eta g_Y(\eta)\,\mathrm{d}\eta$$ and $$u_y^2 = \int_{-\infty}^{\infty} (\eta - y)^2 g_Y(\eta)\,\mathrm{d}\eta$$ where $$g_Y(\eta) = \int_{-\infty}^{\infty} \delta[\eta - f(\boldsymbol{\xi})] g(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}$$ and $\delta(\cdot)$ is the Dirac-delta function
Signature	$[y, u_y] = \mathrm{EvaluateUnc}(N, f, g)$
Properties	None
References	1
1	BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP and OIML, *Guide to the expression of uncertainty in measurement (GUM) (GUM 1995 with minor corrections)*, Bureau International des Poids et Mesures, JCGM 100:2008
Additional notes	2
1	implementations of this computational aim include the law of propagation of uncertainty and a Monte Carlo method
2	analytical solutions are available for some special cases
History	created 2012-11-05 by Peter Harris (NPL, UK)

7 Example 3

This example is concerned with specifying the computational aim for a task, viz., evaluating a key comparison value, its associated standard uncertainty and degrees of equivalence from measurement results provided by the laboratories participating in the comparison, that is interdisciplinary and underpins measurement traceability at the international level (between national metrology institutes). The example illustrates a computational aim for which there are input parameters that take the form of characters or strings and booleans. A set of refinements of the computational aim might be specified in order to address key comparison data evaluation within different metrology areas, such as length, mass and amount of substance.

Unique identifier	en/-/0/000003
Language	English (en)
Title	evaluate reference value, associated standard uncertainty and degrees of equivalence from key comparison data
Keywords	5
1	mutual recognition arrangement

2	key comparison
3	reference value
4	degrees of equivalence
5	weighted mean
Mathematical area	-
Dependencies	None
Input parameters	5
1	N
	number of laboratories
	$\mathcal{N}_0$
	scalar
	$\{N : P(N)\}, \quad P(N) = (N > 0)$
2	$\boldsymbol{L} = (L_1, \ldots, L_N)^\top$
	identifiers L_i for the laboratories (e.g., 'NPL', 'PTB', etc.)
	$\mathcal{S}$
	vector of length N
	$\{\boldsymbol{L} : P(\boldsymbol{L})\}, \quad P(\boldsymbol{L}) = ((\forall\, i,\ \mathrm{len}(L_i) > 0) \wedge (\forall\, i \neq j,\ L_i \neq L_j))$
3	$\boldsymbol{I} = (I_1, \ldots, I_N)^\top$
	those laboratories for which I_i is 'true' are included in the calculation of the reference value and its associated standard uncertainty
	$\mathcal{B}$
	vector of length N
	$\{\boldsymbol{I} : P(\boldsymbol{I})\}, \quad P(\boldsymbol{I}) = (I_1 \vee \cdots \vee I_N)$
4	$\boldsymbol{x} = (x_1, \ldots, x_N)^\top$
	measured values x_i provided by the laboratories
	$\mathcal{R}$
	vector of length N
	none
5	$\boldsymbol{u}_x = (u_1, \ldots, u_N)^\top$
	standard uncertainties u_i associated with the measured values provided by the laboratories
	$\mathcal{R}$

	vector of length N
	$\{\boldsymbol{u}_x : P(\boldsymbol{u}_x)\}, \quad P(\boldsymbol{u}_x) = ((u_1 \geq 0) \wedge \cdots \wedge (u_N \geq 0))$
Output parameters	5
1	y
	key comparison reference value
	$\mathcal{R}$
	scalar
	none
2	u_y
	standard uncertainty associated with the key comparison reference value
	$\mathcal{R}$
	scalar
	$\{u_y : P(u_y)\}, \quad P(u_y) = (u_y \geq 0)$
3	$\boldsymbol{D}$ with $\boldsymbol{D}_{i,1} = d_i,\ \boldsymbol{D}_{i,2} = U(d_i)$
	degrees of equivalence (DoE): d_i is the value component of the DoE, and $U(d_i)$ is the expanded uncertainty associated with d_i corresponding to a level of confidence of 95 %
	$\mathcal{R}$
	matrix of dimension $N \times 2$
	$\{\boldsymbol{D} : P(\boldsymbol{D})\}, \quad P(\boldsymbol{D}) = ((U(d_1) \geq 0) \wedge \cdots \wedge (U(d_N) \geq 0))$
4	$\boldsymbol{L}_1 = (L_{1,1}, \ldots, L_{1,n_1})^\top$
	identifiers $L_{1,i}$ for those laboratories included in the calculation of y and u_y
	$\mathcal{S}$
	vector of length $n_1 \geq 1$, where n_1 is the number of elements of $\boldsymbol{I}$ that are 'true'
	$\{\boldsymbol{L}_1 : P(\boldsymbol{L}_1)\}, \quad P(\boldsymbol{L}_1) = ((s \in \boldsymbol{L}_1 \Rightarrow s \in \boldsymbol{L}) \wedge (\forall\, i \neq j,\ L_{1,i} \neq L_{1,j}))$
5	$\boldsymbol{L}_2 = (L_{2,1}, \ldots, L_{2,n_2})^\top$
	identifiers $L_{2,i}$ for those laboratories not included in the calculation of y and u_y
	$\mathcal{S}$
	vector of length $n_2 \geq 0$, where n_2 is the number of elements of I that are 'false'

	$\{\boldsymbol{L}_2 : P(\boldsymbol{L}_2)\}, \quad P(\boldsymbol{L}_2) = ((s \in \boldsymbol{L}_2 \Rightarrow s \in \boldsymbol{L}) \wedge (\forall\, i \neq j,\ L_{2,i} \neq L_{2,j}))$
Mathematical model	evaluate $$y = \frac{\sum_{i:I_i} x_i/u_i^2}{\sum_{i:I_i} 1/u_i^2}$$ and $$u_y^2 = \frac{1}{\sum_{i:I_i} 1/u_i^2};$$ evaluate $$d_i = x_i - y, \quad U(d_i) = 2u(d_i), \quad i = 1, \ldots, N,$$ where $$u^2(d_i) = u_i^2 - u_y^2,\ \forall\, i : I_i, \quad u^2(d_i) = u_i^2 + u_y^2,\ \forall i : \neg I_i;$$ evaluate $$\boldsymbol{L}_1 = \boldsymbol{L}(\boldsymbol{I}), \quad \boldsymbol{L}_2 = \boldsymbol{L}(\neg \boldsymbol{I})$$
Signature	$[y, u_y, \boldsymbol{D}, \boldsymbol{L}_1, \boldsymbol{L}_2] = \mathrm{KCDataEvaluation}(N, \boldsymbol{L}, \boldsymbol{I}, \boldsymbol{x}, \boldsymbol{u}_x)$
Properties	1
1	$n_1 + n_2 = N$
References	2
1	BIPM, *Mutual recognition of national measurement standards and of calibration and measurement certificates issued by national metrology institutes*, Technical Report, Bureau International des Poids et Mesures, Sèvres, France, 1999 (Technical supplement revised 2003)
2	M G Cox, The evaluation of key comparison data, *Metrologia*, **39**, 58995, 2002
Additional notes	8
1	the constraint on $\boldsymbol{L}$ is to ensure that no element L_i can be the empty string and no two elements are equal
2	the constraint on $\boldsymbol{I}$ is to ensure that at least one element I_i must be 'true'
3	the constraint on $\boldsymbol{L}_1$ is to ensure that its elements are a subset of those of $\boldsymbol{L}$ and no two elements are equal
4	the constraint on $\boldsymbol{L}_2$ is to ensure that its elements are a subset of those of $\boldsymbol{L}$ and no two elements are equal
5	if $n_2 = 0$, $\boldsymbol{L}_2$ is returned as the empty matrix
6	the summations in the definition of the mathematical model are taken over the subset $\boldsymbol{I}_1$ of the indices $\{1, \ldots, N\}$ of those laboratories included in the calculation of the key comparison reference value and its associated standard uncertainty
7	the symbol $\neg$ used in the definition of the mathematical model denotes the logical operator 'NOT'

8	it is recommended that a statistical test, such as a χ^2–test, is applied to confirm the consistency of the provided laboratory measurement results, comprising measured values and associated standard uncertainties, with the calculated weighted mean
History	created 2012-11-05 by Peter Harris (NPL, UK)

8 Summary and conclusions

It is only possible to verify and validate software when it is known what problem the software is intended to solve or task the software is intended to execute. A statement of a specification for the computational aim of the software is used to set the user and functional requirements for the software developer, that is, to specify what is required of software to be a conforming product, and to provide a basis for the verification and validation of a software implementation. This report has described a generic approach to specifying a computational aim and has illustrated that approach with three examples: one that is typical of requirements arising in dimensional metrology, and two that are interdisciplinary being concerned with measurement uncertainty evaluation and key comparison data evaluation. It has been proposed that (a) the abstract and universal language of *mathematics* is used to provide specifications of computational aims, and (b) *refinement* is used to distinguish as far as possible between the purely mathematical problem that is to be solved, which involves operations on numerical values, and an instantiation of the mathematical problem within a metrology area, measuring system or instrument, for which the numerical values will be associated with quantities with given dimensions, measurement units and (possibly) sets of typical values. A companion report [2] describes the mathematical and ICT (information and communications technology) tools used to capture the specification of a computational aim.

Acknowledgements

The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union.

References

[1] BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP and OIML 2012 *International vocabulary of metrology - Basic and general concepts and associated terms (VIM) (3rd edition, 2008 version with minor corrections)* Bureau International des Poids et Mesures, JCGM 200

[2] Forbes A, Harris P, Lines K, Hoang M 2013 *High-level specification of a software system to capture the specifications of computational aims* EMRP NEW06 TraCIM Deliverable 2.1.3–B

[3] Zeleny V and Linkeova I 2012 *Generic basic terms in key areas of computationally-intensive metrology* EMRP NEW06 TraCIM Deliverable 1.3.1

[4] Harry A 1996 *Formal Methods Fact File* John Wiley & Sons (Chichester, England)

[5] Woodcock J and Loomes M 1989 Software Engineering Mathematics Pitman Publishing

[6] ISO 639-1:2002 Codes for the representation of names of languages – Part 1: Alpha-2 code International Organization for Standardization, Geneva

[7] Boisvert R F, Howe S E and Kahaner D K 1985 GAMS - a framework for the management of scientific software *ACM Trans. Math. Software* **11** 313-355 gams.nist.gov/cgi-bin/serve.cgi

[8] www.bipm.org/en/committees/cc/