Software *Metrics* Need to Mature into Software *Metrology* (Recommendations)

Position paper prepared by

Alain Abran

Software Engineering Management Research Laboratory

Université du Québec à Montréal

E.mail: abran.alain@uqam.ca

Abstract:

If software engineering is to mature into a recognized engineering discipline, it needs to be supported by measures, measurement methods and well tested descriptive and quantitative models. Other disciplines have developed a considerable body of knowledge with respect to measures, measurement instruments and quantitative models using measurement results to analyze relationships across objects and attributes. How does software engineering compare to other fields in this respect?

This position paper highlights some current high-level ambiguities in the domain of software metrics, whereas this term is often interpreted with what would be considered multiple definitions; similarly for the expression metrics validation which is used in many ways with different meanings, leaving practitioners confused and researchers with considerable challenge in leveraging other researchers' contributions on similarly named but distinct issues. To reach maturity, the software engineering knowledge are, referred to as software metrics must mature into software metrology as in other disciplines.

This position paper concludes with recommendations for paths to be explored in order to tackle this issue with contributions from the metrology discipline.

Keywords: Metrology, Software Measurement Methods, Software Metrics, Metrics Validation.

NIST Workshop on

Advancing Measurements and Testing for Information Technology (IT)

Gaithersburg (Maryland), October 26-27 1998

United States Department of Commerce

National Institute of Standards and Technology (NIST)

Software *Metrics* Need to Mature into Software *Metrology* (Recommendations)

Position paper prepared by
Alain Abran

1. INTRODUCTION

In the field of software, technology introduction is proceeding at breakneck speed, a vast array of alternatives are being rushed to the market place and researchers are pursuing new knowledge in all directions.

This diversity is not restricted to software tools, but extends to software development methods and activities as well, and even to concepts and models. This diversity of offerings is a characteristic of emerging technologies which are in the immature phase of their life cycle: while innovative, they are usually neither industry-robust nor fully functional, and they need a substantial amount of fine-tuning to fit into an existing infrastructure of more mature technologies.

At this early stage of technological evolution, innovators have their own views on what features are to be expected and desired in proposed solutions. Furthermore, they all propose their own particular solutions and use their own glossaries of terms. And, if the problem is important enough, there is a plethora of innovators.

The field of software *metrics* has many of the characteristics of emerging technologies. If we take, for example, the issue of the measurement of complexity in software code, Zuse [1] had already found in the literature more than 100 *metrics* of code complexity by 1991, and, in the relatively new domain of object-oriented analysis and programming, there were already more than 150 proposed metrics in 1995 [2]; most of these *metrics* are based on the individual models of the various researchers. While this volume of innovation is impressively abundant, questions must be asked about the validity of each *metric* and its success in addressing the issues at hand.

All science and engineering disciplines have developed a considerable body of knowledge with respect to measures, measurement instruments and quantitative models using measurement results to analyze relationships across objects and attributes being measured. In the standardization process of the measurement instruments within all these disciplines, a common body of knowledge has evolved and enjoys a large consensus internationally, over time, within and across disciplines. This body of knowledge is referred to as *metrology*.

Similarly, if software engineering is to mature and be recognized as an engineering discipline, it needs to be supported by measures, measurement methods and well-tested descriptive and quantitative models. In software engineering, there is no corresponding body of knowledge referred to as *metrology*, as is apparent from the NIST 1997 White Paper[3]. When metrology-related concepts are identified in the software engineering field, a few of them are found in a knowledge area referred to as software *metrics* [3]. However, even though the expression *metrics* is most common in software engineering, it must be observed that in science and engineering disciplines there is no generic term and knowledge area like it[1].

Although the number of publications on software *metrics* is quite large, this does not mean in any way that it is a mature knowledge area. Since software *metrics* purports to address the same domain of knowledge as the metrology field, there is indeed a need to investigate the field of software *metrics* from a *metrology* viewpoint in order to identify its strengths and weaknesses by comparing related concepts and methods and evaluating their completeness.

The position taken here is that to reach maturity the software engineering knowledge area referred to as software *metrics* will also need to mature into software *metrology,* as has happened in other disciplines, like the engineering disciplines.

This position paper highlights some current high-level ambiguities in the knowledge area of software *metrics*, whereas this term is often interpreted with what would be considered multiple definitions; similarly for the expression *metrics validation* which is used in many ways with different meanings, leaving practitioners confused when the need arises as to which alternative to select among many, and researchers with considerable challenges in leveraging other researchers' contributions on similarly named but distinct issues.

This paper is organized in the following way. In section 2, the issue of *metrics* is positioned and the measurement methods process model of [4] is presented to position the various interpretations of *metrics*. The third section describes the different types of validation according to which part of the measurement process it refers to: the design of a measurement method, the application of a measurement method or the exploitation of the measurement results (in predictive systems, for instance) and section 4 presents a classification [5] of *metrics* validation proposals from various authors based on the model proposed in [4]. Section 5 positions the NIST White Paper [3] and other work in progress closely related to metrology and presents a set of recommendations on paths to be explored for software *metrics* to mature into software *metrology*.

2. THE RANGE OF OBJECTS REFERRED TO BY THE EXPRESSION *SOFTWARE METRICS*

Over the past twenty years, a significant number of software *metrics* have been proposed to better control and understand software development practices and products. Although

[1] In mathematics, there is a very specialized formula referred to as a 'métrique', but its definition is extremely limiting, and does not assume the tentative generic definition taken for granted in software engineering.

this expression is used extensively in the literature, it does not necessarily have the same interpretation from author to author, and close study of various papers sometimes indicates that within the same text the expression is often used with various and distinct meanings, leaving the reader quite confused as to the precise topics being discussed and investigated.

Similarly, of the significant number of *metrics* proposed, very few have been looked at closely from a metrology perspective. Furthermore, it is currently difficult to analyze the quality of these *metrics* because of a lack of an agreed-upon validation framework.

In [4] this issue was looked at from a perspective closely related, that of *metrology*, that is, from a measurement method perspective. A measurement method process model was proposed which identifies the distinct steps involved, from the design of a measurement method to the exploitation of the measurement results in subsequent models, such as quality and estimation models. These steps are presented in Figure 1:

- **Step 1:** Design of the measurement method: before measuring, it is necessary to design a measurement method.
- **Step 2:** Application of the measurement method rules: the rules of the measurement method are applied to software or a piece of software.
- **Step 3:** Measurement result: the application of the measurement method rules produces a result.
- **Step 4:** Exploitation of the measurement result: the measurement result is exploited in a quantitative or qualitative model.

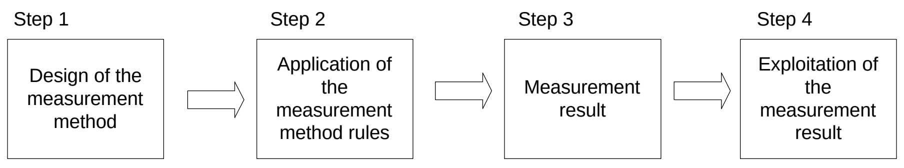
Step 1
Design of the measurement method
Step 2
Application of the measurement method rules
Step 3
Measurement result
Step 4
Exploitation of the measurement result

Figure 1: Measurement Process - High-level Model (Jacquet and Abran 1997 [4])

This high-level model was then refined for the identification of the required substeps within each step.

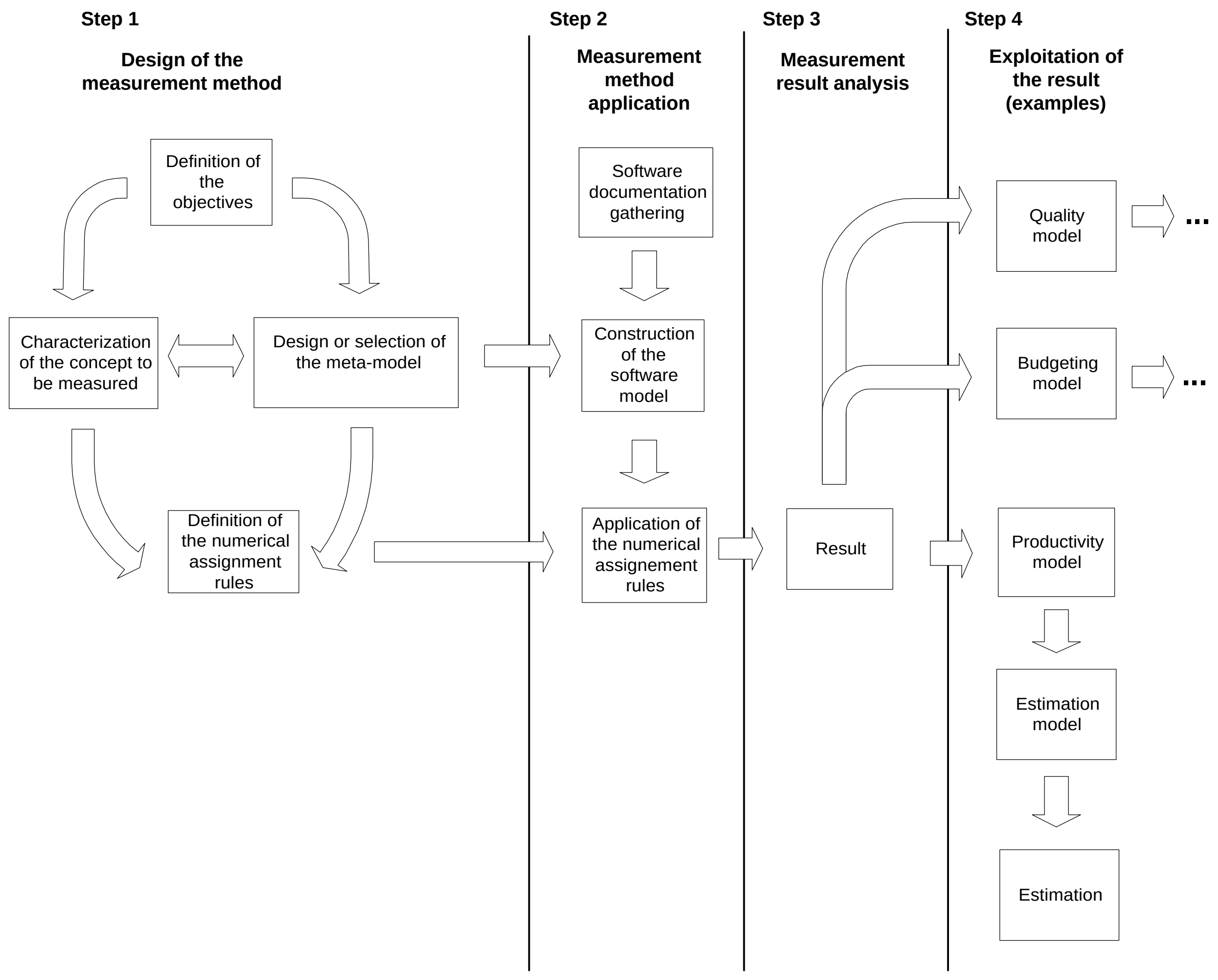
Step 1
Design of the measurement method
Definition of the objectives
Characterization of the concept to be measured
Design or selection of the meta-model
Definition of the numerical assignment rules
Step 2
Measurement method application
Software documentation gathering
Construction of the software model
Application of the numerical assignement rules
Step 3
Measurement result analysis
Result
Step 4
Exploitation of the result (examples)
Quality model
...
Budgeting model
...
Productivity model
Estimation model
Estimation

Figure 2: Measurement Process - Detailed Model (Jacquet and Abran 1997 [5])

The detailed set of substeps identified is illustrated in Figure 2 and these substeps are summarized below:

Step 1: Design of the measurement method

- **Substep 1**: For the initial substep, the objectives of the measurement method must be specified (what object, what attribute, etc.).
- **Substep 2:** Once the attribute (or concept) has been chosen, an (empirical) operational definition of this attribute must be given. This can easily be done for concrete attributes (such as size for a person, or size in lines of code for software), but will be more complicated for abstract attributes (this is what Zuse calls the "*intelligence barrier*" [6]).
- **Substep 3:** Design or selection of the metamodel

  Software is not a tangible product. However, it can be made visible through multiple representations (e.g. for a user, a group of reports, screens, etc.; for a programmer, lines of code, etc.). The set of characteristics selected to represent software or a piece of software, and the set of their relationships, constitute the metamodel proposed for the description of the software to which the proposed measurement method will be applied.

- **Substep 4:** Definition of the numerical assignment rules

  In this substep, the rules allowing assignment of a numerical value to the couple (attribute, object) measured are defined.

Step 2: Application of the measurement method

This step is made up of three substeps:

- Substep 1: the gathering of the documentation artifacts necessary to carry out the application of the measurement method.
- Substep 2: the construction of the software model (for example, the various pertinent entities for the measurement method are referenced, on the basis of the meta-model defined or selected in the design step).
- Substep 3: the application of the numerical assignment rules.

Step 3: The application of the measurement method provides a measurement result.

Step 4: This result of the measurement method can be used in descriptive or predictive models such as quality and estimation models.

3. HOW TO ANALYZE *METRICS* VALIDATION PROPOSALS?

How do you determine that *metrics* are valid? A number of authors in software *metrics* have attempted to answer these questions [7], [8], [9], [10], [11], [12], [13], [14]. An analysis in [5] of the validation-related papers indicates that the validation problem has up to now been tackled from different points of view (mathematical, empirical, etc.) and by giving different interpretations to the expression *metrics validation*; to the point that Kitchenham et al. have suggested that "*what has been missing so far is a proper discussion of relationships among the different approaches*" [10].

The expression *metrics validation* has been used with many distinct interpretations in the software engineering literature, leaving readers somewhat puzzled as to which author's proposals should be used, and under what circumstances. Most authors have not explicitly stated which step of a measurement method process their validation proposal is intended to address. For example, fairly distinct validation methods are proposed for validating *metrics*, but, even though they explicitly refer to this same expression, they do not address the same issue (validation of measurement methods and validation of predictive systems, for instance) and they do not use the same validation techniques.

Rather than trying to recommend and select a single author's interpretation as the correct one for what has been referred to by various authors as *metrics validation*, [5] looked at the validation issue using the measurement process model presented in the previous section. This approach allowed the identification, clarification and positioning of which measurement concepts, and sub-concepts, were being addressed by the various authors. It concurrently identified that there is no single validation methodology for the unique, but ambiguous, expression, *software metrics*. It can be seen that there are indeed different types of components in a measurement method process model, and that each type of component requires a distinct type of validation of their measurement artifacts.

Some classifications of validation studies for software *metrics* have been proposed in the past (see, for example, Gustafson [15]) but these classifications have been made according to different criteria and sometimes without a clear distinction being made between measurement methods and predictive models. Based on the process model presented in the previous section, three major types of validation were identified and are presented next.

3.1 Validation of the design of a measurement method

This type of validation consists in checking that the measurement method really measures what it is supposed to measure. A valid measurement method design would consist of a design for which its numerical assignment rules properly represent the empirical characterization of the measured attribute, and the validation process would consist in proving that empirical data are properly captured by the measurement method design.

It was illustrated in [5] that in the software engineering literature, validation of the design of a measurement method has mainly been tackled from a measurement theory viewpoint: a valid measurement method is a method which verifies the representation theorem. Nevertheless, it appears that this requirement is not sufficient and some authors have proposed additional criteria. However, it seems that the validation of a measurement method has to do with the validation of an empirical relational set, and, consequently, with the validation of attributes and objects (or models of these objects[2]) taking part in the description of this empirical set and consequently in the measurement method. This problem does not seem to have been tackled completely and the validation criteria for this substep of the design of a measurement method are still relatively poorly covered.

3.2 Validation of the application of a measurement method

The second type of validation is the validation of the application of a measurement method: it deals with a specific utilization of a measurement method. Once a measurement method has been validated, it can be applied. But, after the method has been applied, how can it be ascertained that the application process has been properly carried out? What degree of confidence can one have in the measurement result, knowing that mistakes may have been made when applying the method? How can a result be formally accepted before it is used by an organization?

[2] In this, one can find the type of argument made by Gustafson et al. in [15] about the importance of modeling the object to be measured.

Even though these questions are important, they are rarely discussed in the software engineering literature, nor investigated in software organizations, in particular in software measurement groups whose members are often trained from the classic software *metrics* track.

Validation of the application of a measurement method should involve both steps 2 and 3 of the measurement process described in Figure 2 [11].

3.3 Validation of descriptive and predictive models

The third type of validation is the validation of descriptive and predictive models. Schneidewind writes that *if metrics are to be of greatest utility, the validation should be performed in terms of the quality function (quality assessment, control and prediction) that the metrics are to support* [13]. However, Kitchenham et al. disagree with Schneidewind and remark that *validating a predictive or usage model is different from validating a measure* [5]. It is then important to make a real distinction between validation of a measurement method and validation of a predictive system. For example, a predictive model using one or multiple measurement methods can be valid even though one or more of the measurement methods are invalid. This would mean that the measurement methods and the model are self-correctors (i.e. the errors of one counterbalance the errors of the other).

Furthermore, validation of a measurement method is less context-dependent than the validation of a predictive system. For example, one can validate a functional size measurement method for a type of software. Now, this measurement method can be used in a predictive system in order to predict productivity for a specific group of programmers in a specific environment. This predictive system can be validated for this context, but would have to be revalidated if used in another environment. This revalidation involves reexamination of the results of the predictive model in this new context and not the revalidation of the measurement method itself. Consequently predictive systems are validated according to a context.

3.4 Framework of validation types and criteria

In the previous sub-sections, a framework has been explicitly stated which characterizes three types of validation when addressing the validation issue: validation of the design of a measurement method, validation of the application of a measurement method and validation of the use of measurement results in a predictive system.

The proposed process model for software measurement methods is then used to classify various authors' validation criteria according to these measurement process steps. This positioning enables the identification of convergence or dissimilarities among the various validation approaches. A summary of this classification is presented in the appendix. It highlights that, because none of these validation approaches proposed to date in the literature covers the full spectrum of the process of measurement methods, a complete and practical validation framework does not yet exist.

4. NEXT STEPS

4.1 NIST White paper: Metrology for Information Technology

The NIST White Paper [3] has presented a review of the current state of the art in terms of generally accepted measurement concepts in mature disciplines, This body of knowledge, related to measurement concepts and practices, is referred to as *metrology*. In addition to the generally accepted definitions and concepts from the *metrology*, the NIST White Paper has illustrated the logical relationships among the metrology concepts for use in standardization in measurements.

In the context of the need for a mature measurement knowledge area to support the evolution of software engineering towards a status of a mature engineering discipline, metrology-related concepts should be explored and investigated as a promising basis for the structuring of knowledge in our domain of concern, that is, for the design of valid software measurement instruments and their proper use in information technology (IT).

In software engineering the concepts of metrology and its associated rigor in this discipline should not be rejected out of hand because the software artifact, from some viewpoints, appears to be non-physical, and that a new, and immature, discipline of software *metrics* is acceptable over the long term. Related assumptions should be investigated, starting with the identification of the missing components of software *metrics* with respect to the recognized domain of knowledge of *metrology*.

Even though many *metrology* requirements are still missing in the field of software metrics, such as lack of attention to units of measurement, realization of references and traceability, these should be investigated and tackled since such concepts have proven to be critical to the pursuit of advancing knowledge in other domains of scientific disciplines (for understanding both physical and behavioral phenomena).

From the identification of the weaknesses of the current interpretation of both software *metrics* and *metrics validation*, from the measurement process model discussed in sections 3 and 4 above, and from the metrology concepts highlighted in the NIST White Paper, we are now in a position to identify and recommend investigative steps that should be undertaken in the pursuit of an evolution of software *metrics* into software *metrology*. These are presented next.

4.2 Validation framework

Sections 3 and 4 presented a discussion on software *metrics* and the associated concept of a software *metrics validation* and proposed the perspective of a process model of a measurement method to help clarify and position their underlying conceptual components. This work was never intended to be either complete and definitive. A full validation framework is still required, and this problem could perhaps be addressed by investigating validation frameworks from other research fields, such as metrology, the social sciences or the management sciences, for example.

For instance, knowledge from the field of metrology should be investigated to broaden and ensure completeness of a validation framework of measures in the discipline of software engineering. This should help to strengthen significantly this area of measurement validation, to recognize the relevant validation types and to select validation techniques appropriate to each validation type.

- Recommendation 1: Using the body of knowledge of metrology, review the current state of knowledge on *metrics validation* and identify potential contributions for a full validation framework.

4.3 Analysis of current ISO work in progress

Even after 20 years of research in software *metrics* and considerable focus on this issue over the past twenty years, it is surprising that there still do not exist any metrology-related standards in software measurement.

It is indeed challenging to develop a standard in a domain of knowledge that is as chaotic as this one is. The domain must be structured first; only then can the standards be modeled and established. If this structuring has already taken place in the industry, or academia, in one or more formats, as a *de facto* standard(s), the ISO standards-setting process consists in obtaining international agreement on a single standard which will be endorsed universally to become a *de jure* standard. The speed of the standards-setting process will depend heavily on the speed of the consensus-building process towards a single *de jure* standard.

If structuring has not yet taken place, such as is currently the state of the art and practice in software measurement, the ISO working groups face the challenge of defining the vocabulary and the components, as well as the architecture. These are necessary steps in structuring any knowledge area that experts tackle in ISO working groups.

There exists currently some ISO work in progress tackling some issues of software measurement, in various groups working from their own perspective: product quality, functional size measurement methods and process measurement.

4.3.1 Product quality

ISO/IES SC7 working group 6 (WG6) is looking into the measurement of software product quality. It has defined a model of quality (with quality characteristics and sub-characteristics), and is now attempting to select a significant number of *metrics*, which could easily be in the hundreds. However, this current body of work has had no input from a metrology perspective.

- Recommendation no. 2: The current program of work of ISO SC7 WG6 on software product quality should be urgently investigated prior to its final approval to ensure that it meets the sound requirements of *metrology*, including the definition selected for the term *metrics* .

4.3.2 Functional size

ISO/IEC SC7 working group 12 (WG12) looking into functional size measurement methods, has taken a significantly different approach from that of WG6. This working group initially looked at a single such measurement method, but recognized rapidly that they needed to look at it in a much more structured way: the final approach selected was as follows: instead of working at recognizing a single measurement method which did not seem to work outside its initial domain of applicability, the group needed to investigate the appropriate requirements of functional size measurement and of measurement methods. This working group is currently tackling some of the closely related metrology issues, such as validity and verification criteria for a measurement method, including traceability, classification of domains of application and the development of either a reference model or a reference method.

- Recommendation no. 3: The current program of work of ISO SC7 WG12 on functional size measurement methods should be investigated prior to its final approval to ensure its consistency with the requirements of a *metrology* discipline.
- Recommendation no. 4: This current program of work of ISO SC7 WG12 should be investigated to learn how the metrology-related concepts are being tackled, and to identify and document lessons learned.

The lessons learned could facilitate technology transfers to other areas of software measurement methods.

4.3.3 Measurement program

ISO/IEC SC7 working group 13 (WG13) is tackling the definition of the measurement process program framework and is at the project initiation phase. It would therefore be urgent that metrology-related concepts be integrated up-front in its program of work to maximize impact and benefits.

- Recommendation no. 5: The program of work of WG13 on the software measurement process program framework should be investigated up-front to ensure that it will take into consideration the sound requirements of *metrology*, including the definition selected for the term *metrics* and the requirements of traceability.

4.4 Selection of test cases for studying metrology maturity of software measures.

The current body of knowledge on software *metrics* is currently so diversified and has so many proposed alternatives for each and every topic being investigated by researchers, that it is not seen to be economically viable to investigate each of the hundreds of alternative *metrics* proposed to date. It could be of interest to select and analyze two distinct types of *metrics* as test cases for investigating how *metrology* concepts are applied currently in the design of software measures, and where criteria are being met and, when they are not, why not.

Two different types of measures should be investigated: one from the main track, such as a 'complexity metric' or an 'object metric', and one from a non-traditional track, such as a functional size measurement method.

- Recommendation no. 6: Study from a metrology perspective a measure from the software *metrics* track.
- Recommendation no.7: Study from a metrology perspective a measure from the non-classical track, such as Function Point Analysis (or one of its derivative designs for other domains of applicability).

4.5 Measurement unit of information

The NIST white paper has indicated that there *appears to be no recognized, established dimensioning system or quantities relevant to IT metrology. Of the seven base units in SI, only the "second' for time, appears essential for IT metrology. Possibly, the only other base unit necessary for IT metrology is the "bit" of information… Possibly developing such an equivalent would be useful, maybe not.*

However, the following statement needs to be challenged: *One advantage in IT metrology appears to be that, whatever base and derived units are used, the technological challenge posed in realizing SI units does not exist. In other words anyone can define and establish a "bit' of information without use of a measurement device. Possibly all that is needed to define the quantity of information is reference to a classic work, such as Mathematical Theory of Communication by Shannon and Weaver. Such work preceded … but still may sufficiently characterize information as a quantity and bit as a unit of measure".*

Over the past few years, very limited research has been initiated to expand the meager domain of knowledge for the measurement of units of information. However, a potentially closely related measurement method has received a significant amount of attention over the past twenty years, that is the Function Point Analysis method, the purpose of which is to measure the problem to be addressed by software rather than its solution.

It could be of interest to initiate a research project to investigate how metrology concepts, as described in the NIST White Paper on IT Metrology, are presented in such a measurement method, to investigate it from this new (for this measurement method) perspective of the measurement of information, and then to define a strategy to address the missing as well as the ill-defined components.

Although Function Point Analysis (FPA) as a software measurement method is far from perfect, it could be that, from a metrology viewpoint, it is more mature than many other software *metrics* based on software code; FPA's in-depth analysis from a metrology viewpoint could provide useful insights into the domain of software measurement. On the one hand, the lessons learned from the analysis of the FPA measurement method from a metrology perspective could help define a process for reviewing other types of software code *metrics*, thereby providing a foundation for identifying their weaknesses and, consequently, a foundation for systematically improving them. On the other hand, the lessons learned from an analysis of the concepts underlying the phenomenon that Function Point Analysis attempts to measure could provide additional insights into the review of current work on the measurement of information and, perhaps, provide clues for the definition of units of information.

- Recommendation no. 8: Investigate Function Point Analysis (or one of its derivatives) to verify to what degree (in terms of criteria) it meets the *metrology* concepts, with its strengths and weaknesses, and collate lessons learned for the field of software measurement.
- Recommendation no. 9: Investigate Function Point Analysis (or one of its derivatives) to verify whether or not it can contribute to further analysis aimed at the development of measurement of information.

5. SUMMARY

This position paper has highlighted some of the current ambiguities in the knowledge area of software *metrics*, where this term is often interpreted with what would be considered multiple definitions; similarly for the expression *metrics validation,* which is used in many ways with different meanings, leaving practitioners confused and researchers with considerable challenge in leveraging other researchers' contributions on similarly named but distinct issues.

This position paper has also referred to an analysis of the validation approaches proposed in the literature and the basis for this analysis was a process model for software measurement methods which identifies the distinct steps involved from the design of a measurement method to the exploitation of the measurement results. This process model for software measurement methods is used to position various authors' validation criteria according to the measurement process to which they apply. This positioning enables the establishment of relationships among the various validation approaches. It has also made it possible to show that, because none of the validation approaches proposed to date in the literature covers the full spectrum of the process of measurement methods, a complete and practical *metrics validation* framework does not yet exist.

If software engineering is to mature and be recognized as an engineering discipline, it needs to be supported by measures, measurement methods and well-tested descriptive and quantitative models. Other disciplines have developed a considerable body of knowledge with respect to measures, measurement instruments and quantitative models using measurement results to analyze relationships across objects and attributes being measured. How does software engineering compare to other fields in that respect? To reach maturity, its knowledge area, referred to as software *metrics,* also needs to mature into software *metrology* as in other disciplines.

It was therefore proposed that ways and means be investigated to replace current ambiguous software *metrics* terminology with the appropriate *metrology* vocabulary and to use methods from mature disciplines for the design and full validation for both the design and usage of measures and measurement instrumentation.

This position paper has concluded with the following set of recommendations for paths to be explored in order to tackle this issue:

- Recommendation 1: Using the body of knowledge of metrology, review the current state of knowledge on *metrics validation* and identify potential contributions for a full validation framework.
- Recommendation no. 2: The current program of work of ISO SC7 WG6 on software products quality should be urgently investigated prior to its final approval to ensure that it meets the sound requirements of *metrology*, including the definition selected for the term *metrics* .
- Recommendation no. 3: The current program of work of ISO SC7 WG12 on functional size measurement methods should be investigated prior to its final approval to ensure its consistency with the requirements of the *metrology* discipline.
- Recommendation no. 4: The current program of work of ISO SC7 WG12 should be investigated to learn how the metrology-related concepts are being tackled, and to identify and document lessons learned.
- Recommendation no. 5: The program of work ISO SC7 WG13 on the software measurement process program framework should be investigated up-front to ensure that it will take into consideration the sound requirements of *metrology,* including the definition selected for the term *metrics* and the requirements of traceability.
- Recommendation no. 6: Study from a metrology perspective a measure from the classic software *metrics* track.
- Recommendation no.7: Study from a metrology perspective a measure from the non classical track, such a Function Point Analysis (or one of its derivative designs for other domains of applicability).
- Recommendation no. 8: Investigate Function Points Analysis (or one of its derivatives) to verify to what degree (in terms of criteria) it meets the *metrology* concepts, with its strengths and weaknesses, and collate lessons learned for the field of software measurement.
- Recommendation no. 9: Investigate Function Point Analysis (or one of its derivatives) to verify whether or not it can contribute to further analysis aimed at the development of measurement of information.

Acknowledgments

The research which supports this position paper was carried out at the Software Engineering Management Research Lab. at the Université du Quebec à Montréal. This laboratory is made possible through a partnership with Bell Canada. Additional funding is provided by the Natural Sciences and Engineering Research Council of Canada. The opinions expressed in this article are solely those of the author.

Alain Abran
Software Engineering Management Research Laboratory
Department of Computer Science
Université du Québec à Montréal
P.O. Box 8888, Succ. Centre-Ville
Montréal (Québec), Canada
H3C 3P8

E.mail: abran.alain@uqam.ca
Tel: (514) 987-3000 (8900)
Fax: (514) 987-8477

References

[1] H. Zuse, *Software complexity - Measures and methods*. Berlin: Walter de Gruyter, 1991.

[2] T. Fetcke, "Softwaremetriken bei Objektorientierter Programmierung," in *Institut für Quantitative Methoden*. Berlin: Technischen Universität Berlin, 1995, pp. 161.

[3] M. D. Hogan, G. P. Carver, L. J. Carnahan, M. M. Gray, T. H. Hopp, J. Horlick, G. E. Lyon, and E. Messina, "Metrology for Information Technology," National Institute of Standards and Technology - Internal/Interagency Reports - NISTIR, Springfield NISTIR 6025, May 1997.

[4] J.-P. Jacquet and A. Abran, "From Software Metrics to Software Measurement Methods: A Process Model," presented at Third International Symposium and Forum on Software Engineering Standards (ISESS'97), Walnut Creek, CA, 1997.

[5] J.-P. Jacquet and A. Abran, "Metrics Validation Proposals: A Structured Analysis," presented at 8th International Workshop on Software Measurement, Magdeburg, Germany, 1998.

[6] H. Zuse, *A Framework of Software Measurement*. Berlin: Walter de Gruyter, 1998.

[7] K. G. Van der Berg and P. M. Van der Broek, "Axiomatic Validation in the Software Metric Development Process," in *Software Measurement*, A. Melton, Ed.: International Thomson Publishing Company, 1996, pp. 157.

[8] N. E. Fenton and S. L. Pfleeger, *Software Metrics - A rigorous & Practical Approach*, Second ed. London: International Thomson Computer Press, 1997.

[9] N. Fenton and B. Kitchenham, "Validating Software Measures," *Journal of Software Technology, Verification and Reliability*, vol. 1, pp. 27-42, 1991.

[10] B. Kitchenham, S. L. Pfleeger, and N. Fenton, "Towards a Framework for Software Measurement Validation," *IEEE Transactions on Software Engineering*, vol. 21, pp. 929-943, 1995.

[11] P. M. Morris and J.-M. Desharnais, "Validation of Function Points - An Experimental Perspective," presented at IFPUG - Complete the Rings of Measurement... Go for the Gold Using Function Points, Atlanta, Georgia, 1996.

[12] R. E. Prather, "An Axiomatic Theory of Software Complexity Metrics," *The Computer Journal*, vol. 27, pp. 42-45, 1984.

[13] N. F. Schneidewind, "Methodology for validating software metrics," *IEEE Transactions on Software Engineering*, vol. 18, pp. 410-22, 1992.

[14] H. Zuse and P. Boolmann-Sdorra, “Measurement theory and software measures,” in *Formal Aspects of Measurement*, T. Denvir, R. Herman, and R. Whitty, Eds.: Workshops in Computing, Springer-Verlag, 1992, pp. 219-251.

[15] D. A. Gustafson, J. T. Tan, and P. Weaver, “Software Metric Specification,” in *Software Measurement*, A. Melton, Ed.: International Thomson Publishing Company, 1996, pp. 179.

[16] M. Shepperd and D. Ince, *Derivation and Validation of Software Metrics*. Oxford: Clarendon Press, 1993.

[17] R. E. Prather, “An Axiomatic Theory of Software Complexity Measure,” *The Computer Journal*, vol. 27, pp. 340-347, 1984.

[18] E. J. Weyuker, “Evaluating Software Complexity Measures,” *IEEE Transactions of Software Engineering*, vol. 14, pp. 1357-1365, 1988.

Appendix A

Validation types and authors

MEASUREMENT PROCESS MODEL	AUTHORS WHO ADDRESSED THE CONCEPT, AND A FEW EXAMPLES OF VALIDATION CRITERIA
Step 1: Design of the Measurement Method.	
Definition of the objectives.	[10] Kitchenham et al.
Design or Selection of the Metamodel.	Missing
Characterization of the Concept.	[16] Shepperd, Ince, [17] Prather, [18] Weyuker: These authors propose axioms that should be satisfied. These axioms are in the main related to properties of the attribute (or concept) measured. [10] Kictchenham et al.: for example, these authors tackle the attribute validity.
Definition of the Numerical Assignment Rules.	[9] Fenton, [14], [6] Zuse, [10] Kictchenham et al. [16] M. Shepperd, Darrel Ince, etc.: These authors require that the representation condition be satisfied, i.e. that the numerical assignments rules properly characterize the attribute (concept) measured. [10] Kictchenham et al.: for example, these authors tackle the validity of the measurement method unit.
Step 2: Measurement Method Application	
Software Documentation Gathering.	[11] Morris & Desharnais
Construction of the Software Model.	[11] Morris & Desharnais
Application of the Numerical Assignment Rules	[11] Morris & Desharnais
Step 3: Measurement Result Analysis	[11] Morris & Desharnais
Step 4: Exploitation of the Results	[9] Fenton [10] Kitchenham et al. [13] Schneidewind [14] Zuse [6] Zuse