The COSMIC Functional Size Measurement Method

Version 3.0.1

Guideline for assuring the accuracy of measurements

VERSION 1.0

February 2011

Acknowledgements

Version 1.0 authors and reviewers 2011 (alphabetical order)		
Alain Abran, École de Technologie Supérieure, Université du Québec, Canada	Manfred Bundschuh, University of Applied Sciences Cologne, Germany	Jean-Marc Desharnais*, École de Technologie Supérieure, Université du Québec, Canada
Peter Fagg, Pentad Ltd., UK	Arlan Lesterhuis*, The Netherlands	Marie O'Neill, Software Management Methods, Ireland
Grant Rule, Software Measurement Services, United Kingdom*	Luca Santillo, Agile Metrics, Italy	Charles Symons*, United Kingdom
Frank Vogelezang, Ordina, The Netherlands		

* Authors of this Guideline

Version Control

The following table gives the history of the versions of this document.

DATE	REVIEWER(S)	Modifications / Additions
28-2-2011	COSMIC Measurement Practices Committee	First version 1.0 issued

Foreword

Purpose of the Guideline

This Guideline provides an overview of the factors that influence the quality of a functional size measurement, particularly a COSMIC size measurement, and advises how to organize the measurement process so as to obtain accurate size measurements and/or to improve measurement accuracy.

Quality assurance actions aim to improve quality and to help achieve desired quality levels. In our case a measurer's quality aim will be to achieve a desired measurement accuracy level.

The desired accuracy of a measurement (see below for the definition of 'accuracy') will probably vary with the project circumstances. Early in the life of a new software project, only an approximate estimate of size may be possible, but it may still be of acceptable accuracy. At the other extreme, e.g. in the context of outsourced software contracts where a measurement may at some time become a matter of legal dispute, highly accurate measurements are likely to be essential.

In order to assure the desired accuracy of measurements, it is important to have both well-trained and experienced measurers and an adequate and repeatable measurement process. Audits are also necessary to check and to assure on-going accuracy.

In managing the measurement process, two main quality control efforts are essential. They are, in order of importance:

- Error-prevention,
- Defect-detection.

Error-prevention efforts concern three main factors[1], namely assuring the:

- Quality of the measurer(s)
- Quality of the software artifacts
- Quality of the measurement process

Defect-detection that only *identifies* defects may not help much to prevent their re-occurrence. Rather, audits should identify defects *and* determine the causes of the defects that have been made when measuring. Systematically determining the root conditions and analyzing the causes of defects will encourage those responsible to improve the measurement process and in future avoid time wasted in audits and defect corrections.

Intended Readership of the Guideline

This Guideline is intended to be used by measurers and by the management of measurers, who have the task of organizing functional size measurement.

Readers of this Guideline are assumed to be familiar with the COSMIC Method.

Introduction to the contents of the Guideline

Chapter 1 discusses the error-prevention features of the measurement process. Chapter 2 discusses defect-detection, auditing measurements and of the measurement process. Chapter 3 presents a simple audit or 'self-audit' checklist that may be helpful to indicate the quality of a measurement. This

[1] The factors correspond with the sections in chapter 1. The order in which the factors appear does not imply an order of significance.

checklist can be used, for example, by anyone submitting COSMIC-measured project data to the ISBSG repository to give a quality-score for the measurement.

Conventions and terminology used in the Guideline

- Quality control and assurance efforts require a number of actions to be performed. Actions and points of interest are indicated by a '>' sign.
- There are many kinds of documentation and other artifacts used for measuring or estimating functional size, such as functional user requirements (FUR), designs, definition studies or even the software itself (screens, reports, etc). In this Guideline these are all referred to as ‘software artifacts’. Where appropriate, the reader of this Guideline must judge which one (documentation, software or both) applies.
- ‘Measurement accuracy’ is defined as ‘the closeness of agreement between a measured quantity value and a true quantity value of a measurand’[2].
- A ‘defect’ is defined as a product anomaly[3]. A defect is introduced into a work-product when an error or mistake is made in a process.
- ‘Inspection’ is a close examination of something. ‘Auditing’ is the verification of something with respect to a documented procedure. Where there is no documented procedure, read ‘inspection’ where this Guideline mentions ‘audit’.

[2] The ISO-IEC Guide 99: ‘International vocabulary of metrology – Basic and general concepts and associated terms (VIM)’, Latest edition 2007.

[3] From IEEE 982.1-1988 IEEE Standard Dictionary of Measures to Produce Reliable Software.2.00.

Table of Contents

1

ERROR-PREVENTION

1.1 Quality of the measurers

The quality of the measurers is determined by their knowledge of the COSMIC Method and by their experience in measurement. This quality factor pertains to ‘who’ is performing the measurement.

> Measurements may be carried out by a 'measurement team', i.e. a team of dedicated measurers, or by (some of the) developers. However measurement is organized, it is crucial that the measurers can obtain and maintain sufficient experience. For this same reason it is advised against having persons measure infrequently without a strong verification process to support them.

> Measurers should be trained in and if possible should be certified for the COSMIC Method.

> Periodical COSMIC measurement workshops should be performed for refreshing measurement knowledge and exchanging experience. Periodically requiring all measurers to measure the same piece of software in a ‘blind’ or ‘double-blind test’[4] provides a good way of checking competence, mutual consistency of the measurers and of identifying specific measurement problems.

> Audits of measurements should be performed periodically in order to identify conformance to the mandated process, defects arising and variations in approach (see chapter 2).

> Measurers should regularly check on www.cosmicon.com for the availability and relevance of ‘Method Update Bulletins’ (MUB’s) and any discussion of measurement issues.

1.2 Quality of the software artifacts

The quality of the artifacts to be measured is one of the important factors which determine the accuracy of the size measurement. Given the range of artifacts that may have to be used for measurement, from diagrams, to text, to installed software, it is impossible for the COSMIC method to define procedures for the extraction of the Functional User Requirements needed for measurement. This quality factor pertains to ‘what’ is the subject of the measurement, i.e. the information source(s) of the measurement.

The authors of software artifacts may not know which aspects must be made clear for COSMIC sizing. Therefore the following recommendations are made to help obtain good quality artifacts.

> Authors of software artifacts and those who will use them should be knowledgeable about the generalities of Functional Size Measurement and its uses and, at least in outline, knowledgeable about the COSMIC Method. The organisation will benefit from such an investment in know-how because estimates and measurements will be made earlier, more consistently, be more repeatable, with fewer defects, necessitating less rework. (Experience shows that artifacts that cannot be measured almost certainly have defects such as omissions, ambiguities, etc.

[4] In a ‘blind’ test the measurers do not know that a test measurement is being held, the distributor of the measurement does. In a double-‘blind’ test, neither the measurers nor the distributor of the measurement know that a test measurement is being held.

Measuring using the COSMIC method therefore provides a valuable quality control of the software artifacts. See [12]).

> Local instructions should be issued to advise authors how to express software artifacts in a form that is suitable for COSMIC size measurement and/or to adapt software artifacts for measurement purposes. In particular, it should be easy to identify in the software artifacts the central concepts of the COSMIC Method. A few simple examples of what the measurer means with these concepts should encourage authors to capture the required aspects. The measurers should offer assistance to the authors in capturing these aspects, if desired.

> Controlled tests have often shown that the most common errors made by measurers are to miss (i.e. fail to recognise) functionality that should be measured, so that measured sizes are often less than the actual size. Inexperienced measurers miss more functions than experienced measurers. Measurers therefore should check particularly that the software artifacts they are given represent the best knowledge available at the time and are as complete as possible.

> Take particular care early in the life of a software product when the requirements are not yet known in all the detail needed for an accurate measurement and when they may not be stable. When software artifacts are defined coarsely, at a 'high level', approximation variants of the standard COSMIC method may be used (see the 'Advanced and Related Topics' document [3], Chapter 1, available from www.cosmicon.com.). **At all times, but especially early in the life of a software product, it is valuable to estimate and to report the uncertainty in a measured size. This is achieved by expressing the estimate as a range.**

EXAMPLE 1. A range can be determined for instance as follows. Given a software artifact, identify the parts a) that can be measured accurately and b) that cannot be measured accurately. Measure the size of the former parts, resulting in say S CFP. Of the latter parts, estimate the lower and the upper limit of the size (say L respectively U CFP). The true size is then expected to be between S+L and S+U CFP.

EXAMPLE 2. A software contract specifies that at a defined stage early in the life of a project, the functional size must be measured to an accuracy of 'within + or – 20% of the true value'. A measurer is given the task of measuring a certain statement of FUR and concludes that their quality is such that it is not possible to achieve an accuracy of better than + or – 30% of the true size. Hence, the measurer refers the matter to the customer and/or project manager so that either the contractual terms can be adjusted, or the quality of the FUR can be improved.

EXAMPLE 3. An example statement of FUR is produced to be as clear and unambiguous as possible, for use in a certification examination for candidate measurers. The examination board commissions a group of experts to measure the FUR, and to take the mean of their respective sizes to be the 'expected answer' (the nearest the experts can agree on a 'true size'), which happens to be 260 CFP. Taking into account the variation in the measurements made by the experts and the time pressures of an examination, the examination board agrees that any candidate providing a size within the range 250 to 270 CFP will be considered to have produced a sufficiently accurate answer and will pass this test.

> Checks of the quality of a software artifact should be made. See Appendix A for checklists.

1.3 Quality of the measurement process

A document describing an organization's measurement process should provide both an overview to the organization and the detail needed by measurers. This quality factor pertains to 'how' the measurements should be performed.

> Design and describe the desired measurement process. Aspects to be considered are

- The policies for measurement, i.e. purpose of measuring, stakeholders and their benefits.
- The organization of the measurement process: its activities, who initiates, who is involved, expected effort of the involvement, what data is registered, which artifacts produced for whom, and their expected reaction
- At what stage(s) of the development process measurements are required, on which work-products, how they will be used, the minimum standards expected for their accuracy and whether and why measurements are mandatory

- Local variations on the standard COSMIC measurement process, if applicable.
- The steps of each measurement activity and the tools and facilities to be used, with an explanation
- The allocation of responsibilities for measurement data capture and its quality, the organization, maintenance and use of the repository
- The intended uses of the measurement data, e.g. for improving organizational performance, estimating, etc. Requirements for the presentation of the results of measurements, respecting the interests of the various stakeholders in the measurement activities

> As there are many possible ways in which requirements are expressed, there are no general rules for mapping software artifacts to the COSMIC concepts. Therefore, the way of mapping the local types of software artifacts should be documented.

> For each measurement, a Measurement Report should be drawn up. This can be very simple, but should capture the Measurement Strategy parameters as well as the measurement results. Also, measurers should record defects and ambiguities found in the software artifacts as well as any assumptions made from the artifacts. Such reports will serve as the basis for audits and for future measurement process improvement activities. For details, see chapter 5, 'Measurement Reporting', of the Measurement Manual [1].

> Data concerning the measurement process are a source for improvement. Therefore results of the COSMIC measurements should be adequately registered. For details, see section 'Archiving COSMIC measurement results' in chapter 5, 'Measurement Reporting', of the Measurement Manual.

> Measurement data may be stored either centrally (one central measurement data repository) or de-centrally (a number of local collections of measurement data). It must be considered carefully how to organize the storage of the measurement data, taking into account future use of the data. If data are stored de-centrally, procedures must ensure compatibility of the data across the de-central repositories.

> For each measurement, the Measurement Report and the related software artifacts should be stored in such a way as to preserve their relationship. These documents serve as 'witnesses' of the measurement and enable audits afterwards. They establish the audit trail that is beneficial during subsequent enhancement and support and maintenance activities.

> The measurement process should aim to ensure traceable version control of measurement (input) documents. For instance, to ensure that the latest available documents are used for the final Measurement Report, rather than preliminary documents.

> Several tools exist to support the measurement process, ranging from simple spreadsheets to more sophisticated tools for registering measurement data and relating them to the software artifacts used (see www.cosmicon.com for examples). More sophisticated tools exist that combine capturing measurement-data and project planning or for capturing functional user requirements which can be measured automatically. However, the quality of these tools should be verified, especially the quality of non-transparent tools. No tool should be used that is not fully understood by its users. Otherwise, defects of the tool itself and errors made during use of the tool will not be noticed.

> When a (supposed) shortcoming of the COSMIC Method has been encountered, a Comment or Change Request should be submitted as per the process described in Appendix B. The editors of the COSMIC Method will review the Comment or Change Request and issue a solution to a shortcoming, if appropriate, e.g. a Method Update Bulletin, as quickly as possible.

2

AUDITING: DEFECT-DETECTION

The purposes of auditing can be

- to determine the accuracy of a particular measurement, by revealing defects
- to determine how well the measurer has conformed to the measurement process, and to identify opportunities to improve the measurer's skill, experience and inter-measurer consistency
- to verify and validate the measurement process, its effectiveness, cost and timeliness, so that it can be improved

It is common for defects in a measurement to have been caused by defects in the software artifacts, and/or by errors by the measurer in mapping from the artifacts to COSMIC concepts. For example, vagueness or lack of clarity of the software artifacts may lead the measurer to make wrong assumptions and interpretations (implicit or explicit), leading to defects in the measurement. Whatever the cause may be, a very important point is that with a well-organized process for recording and processing any defects found, the *causes* of defects can be analysed, enabling the organization to take action to prevent future defects of similar type, in priority order of their importance and/or frequency. Besides, it enables both auditors and measurers to judge themselves and identify their strong and weak points.

Research has shown that performing functional size measurement itself can be helpful in detecting defects in software artifacts, by revealing omissions or inconsistencies in functional processes that are difficult to detect with regular audit procedures [14].

See also the Checklists in Appendix A which are designed to assist defect detection.

2.1 Auditing of measurements

A complete audit of a particular measurement may have three components.

- An audit of *methodological correctness*: it is usually carried out by a fellow-measurer.
- An audit to verify the *accuracy* of the measurement of the given software artifacts; it is usually carried out by the author of the artifact or a system expert, working with the measurer to explain COSMIC details.
- An audit of the *measurement documentation*; it is usually carried out by a fellow-measurer and may take no more than a few minutes.

> Methodological correctness. The purpose of the first component is to verify, preferably on the basis of a random sample of the software artifacts[5], that the measurement complies with the requirements of the COSMIC Method. After correction, the measurement should be 'methodologically correct'.

After the first component, the fellow measurer and the measurer should discuss the (supposed) defects. When both agree on some defect[6], its cause can be analysed. This information is

[5] Do not use a sample of the *measured* functionality, as in that case functionality that has been forgotten to be measured will not be detected.

[6] Check the (supposed) defects found by the auditor, an auditor can also make mistakes!

captured. When auditor and measurer differ in opinion about the cause of the defect, both causes should be recorded.

The defects found and their causes should be registered and may be categorized as shown in the table below. The findings and a Pareto analysis of the results (i.e. the causes in decreasing number of errors per category) should be presented periodically to the measurers and other stakeholders and should be the basis of recommendations and/or actions.

Defect Cause (to be entered by the auditor)	**Explanation**	**Possible action to prevent re-occurrence**
Measurer	The measurer knows how the situation should have been measured but made a mistake, or the measurer doesn't know how to measure a situation although the COSMIC Method deals with it	Periodical workshops (see section 1.1) and/or train the measurer
Software artifact	The software artifact describes a situation admitting of more than one interpretation, perhaps leading the auditor and the measurer to different solutions	Require the author to correct the defect and re-measure. Draw up requirements for software artifacts to be measurable
Measurement process	Any aspect of the measurement process, (e.g. a local rule for mapping from FUR to COSMIC concepts admits of more than one interpretation)	Depends on the aspect. If the cause is the local set of mapping rules, make the necessary improvements

> Measurement accuracy. The purpose of the second component is to verify that the measurement faithfully represents the functionality. For this, the measurement should be discussed with the author of the software artifacts that have been measured (or with another functional expert representative of the commissioner of the measurement). Their first task is to ensure that the measured software artifacts are still the correct ones[7]. Subsequently, parts of the artifacts that are not clear to the measurer must be clarified by the author and the measurer's assumptions and interpretations must be verified. Finally, the author must verify that no (change of) functionality has been neglected, nor that functionality has been included in the measurement that is not within the scope. After correction, and when the corrected measurement has been checked by the author, it should be 'functionally correct'. Relevant data can now be registered. Defects found should be recorded as per the table above, actions taken and analysed as per the first component.

> Measurement documentation adequacy. The purpose of the third component is to verify that the measurement is correctly finished. For instance, are data correctly registered, are all relevant documents present, have superfluous documents been removed? It should be carried out by a fellow-measurer at the end of the measurement activities. This component may seem trivial but has been found to be very effective in practice. For, once registered a defect won't be noticed as such anymore, resulting in confusing reports and calculations. Moreover, if a defect is noticed long after registration it is often difficult to reconstruct the original situation and repair it, thus undermining the reliability of the measurement and the output based on the data.

2.2 Auditing of the measurers

Besides the audits described in the previous section, the measurers may expect occasional audits to assess their quality, for a number of purposes. For these audits a well-organized registration and

[7] In practice, sometimes software artifacts have been added, removed or changed without the measurers having been informed.

document storage is indispensable, because it enables the measurers to supply a sample of documents, as specified by the auditors, to be audited and/or re-measured.

> Periodically (once or twice a year) an audit may be held on behalf of the management of the measurers. The object of this audit is to assess the quality of the measurers. Ideally it should be held by an independent and certified third party. The audit consists of re-measuring a sample of the measurements for a selected period. Such audits are especially held in situations of outsourced software development, when payments depend on functional sizes, but may also be held for ’training needs analysis’.

2.3 Root cause analysis of defects

If certain types of defects continue to arise repeatedly, in spite of taking the recommended remedial action, then an analysis of the ‘root cause’ will become necessary to assure the quality of the measurements. Examples might be:

- The artifacts made available for measurement continue to be of poor quality. Action may be needed to improve the organization’s documentation standards (to meet measurement needs) or from higher management to enforce their observance
- The training given to measurers and/or to those who produce documentation needed for measurement may be inadequate, requiring either improvement of the training material (e.g. by adding better case studies) or action by the trainers to improve their presentation skills.

3

DETERMINING THE QUALITY RATING OF A MEASUREMENT

All COSMIC functional size measurements should be accompanied by a quality rating[8]. The rating method presented in this chapter is pragmatic, is quick and easy to use and has been applied successfully in practice. The rating level, which should ideally be at level ‘Good’ or ‘OK’, gives an indication of the probable accuracy of the measurement (see below).

To determine the quality rating of a measurement, use the following steps (by means of the self-audit checklist below):

- Assign a rating of ‘Good’, ‘OK’, or ‘Poor’ to each of the three quality aspects
- The quality rating of the measurement is the **lowest** of the three scores

EXAMPLE. If the ratings of the three aspects are ‘Good’, ‘Good’ and ’Poor’, then the quality rating of the measurement is ’Poor’.

When a functional size measurement is based on either a) an approximate method or b) on a known incomplete software artifact the functional size measurement is assigned a rating ‘Approximate’. Note that measurements with this rating may be suitable for purposes where rough estimates suffice.

3.1 The self-audit checklist

Quality Aspect and Ratings	**System ID:**
1. Quality of the measurer(s) Good = Entry-level certified and >2 years experience in COSMIC measurement or 10,000 CFP measured; measures regularly OK = Entry-level certified AND measures at least once per month* Poor = Beginner OR measures only occasionally* * But if the measurement is audited by an independent, experienced measurer assign ‘Good'	
2. Quality of the software artifacts[9] Good = Excellent documentation; access to the actual system AND to a system expert OK = Reasonable documentation AND access to the actual system OR to a system expert Poor = Limited or poor documentation; no access to the actual system or to a system expert	
3. Quality of the measurement process Good = Measurement records show traceability of the functional processes and data movement types (E, X, R, W) to the software artifacts, on at least a standard spreadsheet; other	

[8] The approach sketched in this chapter is aligned with the rating method used for measurements submitted to ISBSG (International Software Benchmarking Standards Group, a global and independent repository and source of data and analysis for the IT industry), and if used might therefore facilitate the collection, rating and submission of COSMIC-measured project data to ISBSG. ISBSG uses the levels A, B and C with A = ‘Good’, B = ‘OK’ and C = ‘Poor’.

[9] A detailed method of assessing the rating for this aspect is given in section 3.2

		measurement parameters are well documented. There was sufficient time for proper measurement and recording
OK	=	Standard measurement process. The measurement is recorded at least to the functional process level with references to the software artifacts
Poor	=	Informal (undocumented), or individual process, or measurement under severe time pressure; measurement results are not clearly traceable to the software artifacts.

For information

There are no data on how the accuracy of a measurement varies with the quality rating; an initial guess is that the variation is roughly as follows.

Good	= within 5% of the true size
OK	= within 10% of the true size
Poor	= unknown accuracy
Approximate	= depends on the approximation method used

Note. One of the commonest measurement errors is to miss some functionality so that the measured size is lower than the true size. Claiming an accuracy of 'within 10%' may not, therefore, be the same as claiming that the result is 'within plus or minus 10%' of the true size.

3.2 Rating the quality of software artifacts

This section describes a simple process to obtain the rating (Good, OK or Poor) of aspect 2 for a set of software artifacts (the 'software artifacts rating'), in a way that is less subjective than just making a single judgment based on the criteria given in the table above. This process might be applied for instance to support a benchmarking process rating, e.g. if the measure being examined is to be submitted to ISBSG repositories. The software artifacts rating is obtained by requiring the rating to be traced directly to the software artifacts used for the measurement.

The software artifacts rating is determined in two steps. In the first step the functional processes of the software artifacts are rated, leading to a number of functional process ratings (sections 3.2.1 and 3.2.2). In the second step the functional process ratings are used to derive the software artifacts rating (section 3.2.3).

Note. Capturing the functional process ratings at the time of the measurement requires very little extra effort.

3.2.1 Determining the functional process ratings

Rate each functional process identified in the set of software artifacts on the scale (a) to (e) as follows:

(a) The functional process is completely documented

(b) The functional process is documented but the description of the data moved is unclear

(c) The functional process is identified only

(d) The number of the functional processes is given but they are not specified

(e) The functional process is not mentioned in the artifacts but is implicit

Explanation of the functional process ratings:

(a) Each functional process is fully documented, together with its data movements by type (Entry, Exit, Read and Write) including the objects of interest and their attributes.

EXAMPLE of a FUR. 'The system will produce a monthly file. The interface format details are provided in the Interface ID v2.8 in an Excel format document and the entity relationship diagram that shows each attribute is in the ER.gif document version 2.1.'

(b) Each functional process is documented. The input, output, stores and retrievals of each functional process are also described but not clearly enough to identify the number of data movements

EXAMPLE of a FUR. A yearly report will be provided to the manager with the following information: administration category, name of the administration, origin, amount of sales by month, type of product and sales by product, etc. (In this case the reference to the data groups is not clear.)

(c) Functional processes can be identified but not their data movements

EXAMPLE of a FUR. The sales representative will provide a monthly report to the manager.

(d) The number of the functional processes is given but they are not specified.

EXAMPLE of a FUR. Five reports will be provided to the manager.

(e) The functional process is not mentioned in the artifacts but is implicit

EXAMPLE. A screen that allows modification of data about an object of interest within a functional process usually requires the display of the information before modifying the data. Sometimes this display requirement is not documented.

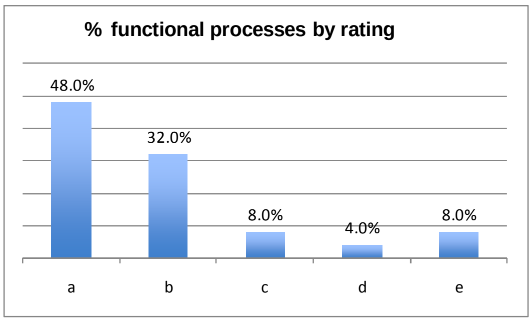
Note 1. A functional process rating may be increased if access to the actual system and/or to a system expert can provide reliable information that is not available in the artifacts.

Note 2. To promote good documentation, it may be helpful to record two functional process ratings: one using the documentation only, and the other taking into account information from an expert and/or from examining the actual system. For ISBSG data submission, only the latter rating should be considered.

3.2.2 Presentation of the functional process ratings

The distribution of the quality ratings for the documentation could be provided based on the percentage of functional processes in each rating. The table shows an example of the rating of a set of software artifacts describing 50 functional processes (from a real set of FUR) and the results are also shown graphically.

Rating	Number of functional processes	Percentage
a	24	48,0%
b	16	32,0%
c	4	8,0%
d	2	4,0%
e	4	8,0%
Total	50	100,0%



3.2.3 Interpretation of the functional process ratings and deriving the software artifacts rating

From these results, the software artifacts rating should be given as ‘Good’, ‘OK’ or ‘Poor’. The following is a suggested way of deriving this quality rating

- A result with more than 70% of the functional processes rated as (a) should be considered for a software artifacts rating of ‘Good’.
- A result of 80% or more of the functional processes rated as (a) or (b), of which at least 50% (a) should also be considered for a software artifacts rating of ‘Good’.
- A result with 60% or more of the functional processes rated as (a) or (b) and at least 30% (a) should be considered for a software artifacts rating of ‘OK’.
- A result with 70% or more of the functional processes rated as (b) and less than 10% rated as (a) should be considered for a software artifacts rating of ‘OK’.
- A result with less than 50% of the functional processes rated as (a) or (b), and less than 10% rated as (a) should be considered for a software artifacts rating of ‘Poor’.

With 5 functional process ratings there are 31 possibilities[10], so the measurer will need to use some judgment when applying these suggestions.

EXAMPLE. The example results given in the table and graph above, with 48% (a) and 32% (b), could lead to a software artifacts rating of

- ‘Good’, applying the suggestion of the second bullet or
- ‘OK’, applying the suggestion of the third bullet

Because 48% is not too far from 50% and the total is 80% the software artifacts rating could be ‘Good’.

The rating method could be further refined by taking into account the size of the functional processes as well as their percentages. So the rating of larger functional processes would carry more weight than smaller functional processes.

[10] The possibilities correspond with the 31 non-empty subsets of the set {a, b, c, d, e}.

LITERATURE

[1] COSMIC, 'COSMIC FSM Method v3.0.1 - Measurement Manual (The COSMIC Implementation Guide for ISO/IEC 19761: 2003)', 2009.

[2] COSMIC, 'COSMIC FSM Method v3.0 - Guideline for Sizing Business Application Software, v1.1', 2008.

[3] COSMIC, 'COSMIC FSM Method v3.0 - Advanced & Related Topics', 2007.

[4] COSMIC, 'COSMIC FSM Method v3.0.1 - Documentation Overview and Glossary of Terms', 2009.

[5] COSMIC, 'COSMIC FSM Method v3.0.1 - Guideline for Sizing Service-Oriented Architecture Software, 2010.

[6] COSMIC, 'COSMIC FSM Method v3.0.1 - Guideline for Sizing Data-Warehouse Software, 2009.

[7] Abran, A. 'Software Metrics and Software Metrology', Wiley, 2010.

[8] Ungan, E, Demirörs, O., Top, Ö.Ö., Özkan, B., 'An Experimental Study of the Reliability of COSMIC Measurement Results', Middle East Technical University, Turkey, 2010.

[9] Desharnais J.M., Abran, A., 'Assessment of the quality of functional user requirements documentation using criteria derived from measurement with COSMIC – ISO 19761'. IWSM/Metrikon conference, Stuttgart, November 2010.

[10] ISO/IEC 15939 - Software Measurement Process.

[11] ISO/IEC TR 14143-3 - Verification of functional size measurement methods.

[12] Rule. P.G., 'Compliant, Correct, Consistent, Complete. UKSMA's Improvement Plan', The Guild of Independent Function Points Analysts, Ltd., International Function Point User Group Conference, Washington, 1998.

[13] Trudel, S., Abran, A., 'Functional size measurement quality challenges for inexperienced measurers', International Workshop on Software Measurement, Amsterdam, Nov. 4-6 2009.

[14] Trudel, S., Abran, A., 'Improving quality of functional requirements by measuring their functional size', International Workshop on Software Measurement, Munich, Nov. 17-19 2008.

APPENDIX A – CHECKLISTS

The three checklists in this appendix can be used together or separately. The first checklist concerns the form and content of the software artifacts. The second checklist can be used to enable a systematic check of measurements. The last checklist focuses on areas where errors are commonly made and may be used when time is limited.

A.1 Checks on the quality of software artifacts

> Do all software artifacts of one organization presented to the measurers (in one measurement or in measurements over time) conform to local standards and have the same format? The same content (i.e. the same way of specifying the events, functional processes, the objects of interest)? The same way of diagramming data models or other concepts? The same level of granularity? Do they describe software at the same level of decomposition? (A lack of consistency of documentation across the organization may be an indicator of quality problems.)

> Does each software artifact describe the purpose of the software and the (business) process that it must support?

> Does each software artifact contain all the information that the measurer needs for the COSMIC measurement (can events, functional processes etc. easily be identified)? Is this information clear and unambiguous?

> Plausibility checks (for instance in a business or MIS application): the CRUDL check relates to the expectation that for each object of interest for which data is stored persistently there will be at least one functional process that Creates (i.e. stores persistently) the data describing the object, one that Reads the data, one (often more than one) that Updates the persistently stored data, one that Deletes the data and one that Lists all instances of the object of interest.

> Cross-checks. Is each functional process identified reflected in the software artifact (i.e. is the software artifact complete)?

A.2 Checks on the quality of measurements

A.2.1 Checklist for the Measurement Strategy Phase

> Is it stated why the measurement is required, what the result will be used for and what is the target accuracy?

> Is the purpose to measure all the delivered software or just the newly developed or enhanced software?

> Is the overall scope stated? Are the scopes stated of the individual pieces of software to be measured separately?

> If within the overall scope the components of a piece of software are distinguished, are their levels of decomposition also identified?

> Does the overall scope include software in more than one layer? If so, is the scope of any one measurement limited to one layer?

> For each piece of software to be measured, are all its functional users identified? Are the functional users in agreement with the purpose of the measurement?

> If the overall scope of the measurement contains several software artifacts, have their levels of granularity been identified? Are the measurements made at the same level of granularity?

> If a measurement was scaled to the level of granularity of functional processes or an approximate sizing method was used, has the accuracy of the resulting size been estimated?

A.2.2 Checklist for the Mapping Phase

> Is each functional process associated with a triggering event and a functional user? Does each identified event trigger at least one functional process?

> Does each functional process represent a view of the software of the functional user that triggered it?

> Does every functional user either detect at least one event, and/or receive data from a functional process?

> Is each functional process indivisible, i.e. there is no part of the functional process that can be triggered by a separate event?

> Is each functional process complete (i.e. does it include all that is required to be done in response to the triggering event)?

> Are the objects of interest identified? Is each object of interest identified really an object of interest to a functional user?

> Is each attribute in the data movements related to exactly one object of interest?

> For each functional process, is it clear which data it has to store persistently or retrieve from persistent storage itself and for which data it must call other software to store or retrieve it?

A.2.3 Checklist for the Measurement Phase

> If a functional process must store data persistently or retrieve data from persistent storage, are one or more Write or Read data movements identified respectively?

> If a functional process must communicate with another piece of software (i.e. a functional user) to store or retrieve data, is one Exit per object of interest identified for the request and one Entry per object of interest identified for the returned data or message?

> Does the data stored persistently for each object of interest get Read by a functional process of the software being measured or by some other software?

> If a functional process must obtain data from a functional user but the latter does not need to be told what data to send, is one Entry and no Exit identified?

> If the FUR require a functional process to output messages without user data (e.g. error messages), is a single Exit identified?

> If data attributes of one object of interest must be entered into a functional process, is one Entry identified (unless the FUR specify otherwise)? In like manner for Read, Write or Exit data movement?

> (In the business application domain only) Are control commands ignored (e.g. navigation commands, page up/down, clicking ‘OK’ for confirmation, etc)?

> Have clock and timing triggering events been identified?

> For an enhancement project, has the analysis identified all data movements that have been added, changed or deleted?

A.3 Checks on areas of commonly-made errors

> Identify assumptions and interpretations, check their validity and check the relevant parts of the measurement, especially with regard to:

- the data model, e.g. an entity-relationship model, a relational data model or a model of object classes
- data that are not attributes of an object (that is normally) of interest, e.g. standard screen headings, data on a report such as page numbers
- parameter tables
- drop-down lists
- objects of interest sub-types
- multiple occurrences of the same type of data movement
- recursive relationships between objects of interest
- re-use of functionality
- omitted ‘list before update’ (i.e. measuring physical not logical screens)
- logon functionality
- drilldown functionality
- the confirmation/error messages Exit
- clock and timing triggering events

> (For functional processes incorrectly combined into one functional process) Identify missing functional processes by verifying that any triggering event corresponds with a functional process

> (For functional processes incorrectly split up into functional processes) Identify redundant functional processes by verifying that any functional process corresponds with one triggering event.

APPENDIX B - COSMIC CHANGE REQUEST AND COMMENT PROCEDURE

The COSMIC Measurement Practices Committee (MPC) is very eager to receive feedback, comments and, if needed, Change Requests for this Guideline. This appendix sets out how to communicate with the COSMIC MPC.

All communications to the COSMIC MPC should be sent by e-mail to the following address:

mpc-chair@cosmicon.com

Informal general feedback and comments

Informal comments and/or feedback concerning the Guideline, such as any difficulties of understanding or applying the COSMIC method, suggestions for general improvement, etc should be sent by e-mail to the above address. Messages will be logged and will generally be acknowledged within two weeks of receipt. The MPC cannot guarantee to action such general comments.

Formal change requests

Where the reader of the Guideline believes there is a defect in the text, a need for clarification, or that some text needs enhancing, a formal Change Request ('CR') may be submitted. Formal CR's will be logged and acknowledged within two weeks of receipt. Each CR will then be allocated a serial number and it will be circulated to members of the COSMIC MPC, a world wide group of experts in the COSMIC method. Their normal review cycle takes a minimum of one month and may take longer if the CR proves difficult to resolve. The outcome of the review may be that the CR will be accepted, or rejected, or 'held pending further discussion' (in the latter case, for example if there is a dependency on another CR), and the outcome will be communicated back to the Submitter as soon as practicable.

A formal CR will be accepted only if it is documented with all the following information.

- Name, position and organization of the person submitting the CR.
- Contact details for the person submitting the CR.
- Date of submission.
- General statement of the purpose of the CR (e.g. 'need to improve text…').
- Actual text that needs changing, replacing or deleting (or clear reference thereto).
- Proposed additional or replacement text.
- Full explanation of why the change is necessary.

A form for submitting a CR is available from the www.cosmicon.com site.

The decision of the COSMIC MPC on the outcome of a CR review and, if accepted, on which version of the business application Guideline the CR will be applied to, is final.

Questions on the application of the COSMIC method

The COSMIC MPC regrets that it is unable to answer questions related to the use or application of the COSMIC method. Commercial organizations exist that can provide training and consultancy or tool support for the method. Please consult the www.cosmicon.com web-site for further detail.