INTERNATIONAL STANDARD

ISO/IEC 19761

First edition
2003-02-15

Software engineering — COSMIC-FFP — A functional size measurement method

Ingénierie du logiciel — COSMIC-FFP — Méthode fonctionnelle de mesure de taille

Reference number
ISO/IEC 19761:2003(E)

© ISO/IEC 2003

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2003

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

© ISO/IEC 2003 — All rights reserved

Not for Resale

Contents

Page

Foreword ... iv

Introduction ... v

1 Scope ... 1

2 Normative references ... 1

3 Terms and definitions ... 2

4 Symbols and abbreviated terms ... 5

5 Unit of Measurement ... 5

6 Measurement activities ... 6
6.1 General ... 6
6.2 Determination of the purpose and scope of the FSM ... 6
6.3 Identification of the FUR ... 6
6.4 Identification of software layers ... 6
6.4.1 Identify software layers ... 6
6.4.2 Characteristics of software layers ... 7
6.5 Identification of software boundaries ... 7
6.6 Identification of functional processes ... 8
6.7 Identification of data groups ... 8
6.8 Identification of data movements ... 8
6.9 Classification of data movements ... 8
6.9.1 Entry ... 8
6.9.2 Exit ... 9
6.9.3 Read ... 9
6.9.4 Write ... 9
6.10 Calculation of the Functional Size ... 10
6.10.1 Assignment of a size unit ... 10
6.10.2 Aggregation of functional size for a functional process ... 10
6.10.3 Aggregation of functional size for the identified FURs for each piece of software to be measured ... 10
6.10.4 Calculation of Functional Size of changes to the FURs ... 11

7 Local customizations ... 11

8 Measurement Reporting ... 11
8.1 Labelling ... 11
8.2 Customization ... 11
8.3 Documentation of the measurement results ... 12

Annex A (informative) **Extraction of Functional User Requirements** ... 13

Annex B (informative) **Allocation of Functional User Requirements (FUR)** ... 15

Bibliography ... 17

© ISO/IEC 2003 – All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19761 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and system engineering*.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

Software is a major component of many corporate budgets. Organisations recognise the importance of controlling software expenses and analysing the performance of the budgets allocated to software development and maintenance in order to benchmark against the best in the field. To do so, measures and models using these measures are needed.

Measures are needed for analysing both the quality and the productivity associated with developing and maintaining software. On the one hand, technical measures are needed to quantify the technical performance of products or services from a developer's viewpoint. Technical measures can be used for efficiency analysis; to improve the performance of designs, for instance.

On the other hand, functional measures are needed to quantify the performance of products or services from a user's or owner's perspective; for productivity analysis, for instance. Functional measures must be independent of technical development and implementation decisions. They can then be used to compare the productivity of different techniques and technologies.

The Full Function Points (FFP) method was proposed in 1997 with the aim of offering a functional size measure specifically adapted to real-time software. Since then, measurement practice in many organisations and field tests demonstrated that this functional size measurement method not only had the ability to capture the functional size of real-time software, but also to capture the functional size of MIS and system software.

These field test results, coupled with the feedback received from organisations which used it, motivated the authors to improve the method. Many improvements were also inspired by the work of the Common Software Measurement International Consortium (COSMIC). The results of these efforts were published in May 2001 as version 2.1 of the COSMIC-FFP Functional Size Measurement Method.

The COSMIC-FFP Functional Size Measurement Method aims to meet these needs of

a) software suppliers facing the task of translating customer requirements into the size of software to be produced as a key activity in their project cost estimating and,

b) customers who want to know the functional size of delivered software as an important component of measuring supplier performance.

© ISO/IEC 2003 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Software engineering — COSMIC-FFP — A functional size measurement method

1 Scope

This International Standard specifies the set of definitions, conventions and activities of the COSMIC-FFP Functional Size Measurement Method. It is applicable to software from the following functional domains:

a) application software which is needed to support business administration;

EXAMPLE Banking, insurance, accounting, personnel, purchasing, distribution or manufacturing.

b) real-time software, the task of which is to keep up with or control events happening in the real world;

EXAMPLE Software for telephone exchanges and message switching, software embedded in devices to control machines such as domestic appliances, lifts and car engines, for process control and automatic data acquisition, and within the operating system of computers.

c) hybrids of the above.

EXAMPLE Real-time reservation systems for airlines or hotels.

This International Standard has not been designed for measuring the functional size of a piece of software, or its parts, which:

d) is characterized by complex mathematical algorithms or other specialized and complex rules, such as may be found in expert systems, simulation software, self-learning software and weather forecasting systems, or

e) processes continuous variables such as audio sounds or video images, such as may be found, for instance, in computer game software, musical instruments and the like.

However, within the local environment of an organisation using the COSMIC-FFP Functional Size Measurement Method, it may be possible to measure these FUR in a way which is meaningful as a local standard. This International Standard contains provision for the local customisation of the method for this purpose.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14143-1:1998, *Information technology — Software measurement — Functional size measurement — Part 1: Definition of concepts*

© ISO/IEC 2003 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply. Whenever a term was already defined by ISO, such as “Functional Size Measurement”, the ISO definition has been adopted for this International Standard.

For some of the terms listed here, the suffix ‘-Type’ is added in brackets. This is to emphasise that, almost invariably throughout this International Standard, for these concepts their ‘Type’ is meant and not their ‘Occurrence’. However, in the main text the suffix ‘Type’ is dropped from these terms for the sake of readability, except when specifically needed to distinguish Type and Occurrence.

3.1
Base Functional Component (BFC)
an elementary unit of Functional User Requirements defined by and used by an FSM Method for measurement purposes

[ISO/IEC 14143-1:1998, definition 3.1]

NOTE The COSMIC-FFP FSM Method defines a data movement as a BFC.

3.2
Base Functional Component Type (BFC Type)
a defined category of BFCs

[ISO/IEC 14143-1:1998, definition 3.2]

NOTE The COSMIC-FFP FSM Method has four BFC Types: Entry, Exit, Read and Write (-Types).

3.3
boundary
a conceptual interface between the software under study and its users

[ISO/IEC 14143-1:1998, definition 3.3]

NOTE The boundary of a piece of software is the conceptual frontier between this piece and the environment in which it operates, as it is perceived externally from the perspective of its users. The boundary allows the measurement analyst to distinguish, without ambiguity, what is included inside the measured software from what is part of the measured software’s operating environment.

3.4
data attribute
the smallest parcel of information, within an identified data group, carrying a meaning from the perspective of the software’s Functional User Requirements

3.5
data group
a distinct, non empty, non ordered and non redundant set of data attributes where each included data attribute describes a complementary aspect of the same object of interest (see 3.17)

3.6
data movement (-type)
A Base Functional Component which moves one or more data attributes belonging to a single data group

NOTE 1 There are four types of COSMIC-FFP data movements: Entry, Exit, Read and Write.

3.7
Entry (-type)
a data movement type that moves a data group from a user across the boundary into the functional process where it is required

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

NOTE 1 In COSMIC-FFP, an Entry is considered to include certain associated data manipulations (e.g. validation of the entered data).

NOTE 2 A data manipulation is anything that happens to data other than a movement.

3.8
Exit (-type)
a data movement type that moves a data group from a functional process across the boundary to the user that requires it

NOTE An Exit is considered to include certain associated data manipulations (e.g. formatting and routing associated with the data to be exited).

3.9
functional process
an elementary component of a set of Functional User Requirements, comprising a unique, cohesive and independently executable set of data movements. It is triggered by one or more Triggering events either directly, or indirectly via an 'actor'. It is complete when it has executed all that is required to be done in response to the Triggering event (-type)

3.10
Functional Size Measurement (FSM)
the process of measuring Functional Size

[ISO/IEC 14143-1:1998, definition 3.7]

3.11
Functional Size Measurement Method
a specific implementation of FSM defined by a set of rules, which conforms to the mandatory features of ISO/IEC 14143-1:1998

[ISO/IEC 14143-1:1998, definition 3.4]

3.12
Functional User Requirements (FUR)
a sub-set of the user requirements. The Functional User Requirements represent the user practices and procedures that the software must perform to fulfil the user's needs. They exclude Quality Requirements and any Technical Requirements

[ISO/IEC 14143-1:1998, definition 3.8]

3.13
layer
the result of the functional partitioning of the software environment such that all included functional processes perform at the same level of abstraction

NOTE In a multi-layer software environment, software in one layer exchanges data with software in another layer through their respective functional processes. These interactions are hierarchical in nature; when considered in pairs, one layer is a "client" to the other. A "client" layer uses the functional services provided by other subordinate layers. Software items in the same layer can also exchange data. This type of data exchange is usually called "peer-to-peer" data exchange.

3.14
measurement method
a logical sequence of operations, described generically, used in the performance of measurements

[ISO, International Vocabulary of Basic and General Terms in Metrology, 1993]

© ISO/IEC 2003 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

3.15
measurement procedure
a set of operations, described specifically, used in the performance of particular measurements according to a given method

[ISO, International Vocabulary of Basic and General Terms in Metrology, 1993]

3.16
measurement process
the process of establishing, planning, performing and evaluating software measurement within an overall project or organisational measurement structure

[ISO/IEC 15939:2002, definition 3.24]

3.17
object of interest (-type)
is identified from the point of view of the Functional User Requirements and may be any physical thing, as well as any conceptual object or part of a conceptual object in the world of the user about which the software is required to process and/or store data

NOTE An Object of interest is a synonym of 'entity-type' on an entity-relationship diagram, and has the same meaning as the subject of a relation in Third Normal Form.

3.18
operating environment (software)
the set of software operating concurrently on a specified computer system

3.19
persistent storage
storage which enables a functional process to store data beyond the life of the functional process and/or which enables a functional process to retrieve data stored by another functional process, or stored by an earlier occurrence of the same functional process

NOTE 1 In the COSMIC FFP model, because Persistent Storage is on the software side of the boundary, it is not considered to be a User of the software being measured.

NOTE 2 Persistent storage enables a functional process to store data beyond the life of the functional process and/or enables a functional process to retrieve data which was stored by another functional process or stored by an earlier occurrence of the same functional process, or stored by some other process, e.g. in the manufacture of a read-only memory.

3.20
quantity (base)
one of the quantities that, in a system of quantities, are conventionally accepted as functionally independent of one another

[ISO, International Vocabulary of Basic and General Terms in Metrology, 1993]

3.21
Read (-type)
a data movement type that moves a data group from persistent storage within reach of the functional process which requires it

NOTE A Read is considered to include certain associated data manipulations necessary to achieve the Read.

3.22
Triggering event
an event that occurs outside the boundary of the measured software and initiates one or more functional processes

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

NOTE 1 In a set of Functional User Requirements, each event-type which triggers a functional process is indivisible for that set of FUR.

NOTE 2 Clock and timing events can be triggering events.

NOTE 3 An event has either happened, or it has not, it is instantaneous.

3.23
unit of measurement
a particular quantity, defined and adopted by convention, with which other quantities of the same kind are compared in order to express their magnitudes relative to that quantity

NOTE Units of measurement have conventionally assigned names and symbols.

[ISO, International Vocabulary of Basic and General Terms in Metrology, 1993]

3.24
users
any person that specifies Functional User Requirements and/or any person or thing that communicates or interacts with the software at any time

[ISO/IEC 14143-1:1998, definition 3.17]

NOTE Users can be human beings, software or engineered devices.

3.25
Value (of a quantity)
magnitude of a particular quantity, generally expressed as a unit of measurement multiplied by a number

[ISO, International Vocabulary of Basic and General Terms in Metrology, 1993]

3.26
Write (-type)
a data movement type that moves a data group lying inside the functional process to persistent storage

NOTE A Write is considered to include certain associated data manipulations necessary to achieve the Write.

4 Symbols and abbreviated terms

BFC: Base functional component

Cfsu: COSMIC functional size unit

FSM: Functional Size Measurement

FUR: Functional User Requirement

MIS: Management Information Systems

Ver. Version (see 8.2)

5 Unit of Measurement

The COSMIC-FFP unit of measurement has the following characteristics:

a) it is defined as one data movement;

b) it is denoted by the symbol Cfsu (Cosmic Functional Size Unit).

© ISO/IEC 2003 · All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6 Measurement activities

6.1 General

The determination of the COSMIC-FFP functional size shall involve all of the activities described in Clause 6.

6.2 Determination of the purpose and scope of the FSM

The purpose and the scope of the Functional Size Measurement shall be determined before commencing a particular measurement exercise.

NOTE 1 It is outside the scope of this International Standard to describe or propose to the measurement analyst specific purposes for performing measurement. Based on the purpose of the measurement, it is up to the measurement analyst to determine the most appropriate artifacts to use to perform a specific measurement.

NOTE 2 The Purpose and Scope of a specific FSM when combined are referred to as the 'Viewpoint'.

EXAMPLE 1 For one piece of software there might be a purpose to measure the functional size as seen by the (human) users of the software, and a separate purpose to measure the size of components which the software development team has to deliver. These two purposes will usually result in different functional sizes. The human user may see only the functionality supplied by the application layer. In contrast, the development team may be required to develop and/or amend software in lower layers of the architecture in addition to developing the application layer, in order to meet the Functional User Requirements and other (non-functional) requirements of the application layer.

EXAMPLE 2 If the purpose is to measure the functional size of software delivered by a particular project team, it will first be necessary to establish the scopes of the various components delivered. These might include software which was used once only to convert data from software which is being replaced. If then the purpose is changed to measure the size which the user has available once the new software is operational, the size measured would be smaller, as the software used for conversion would not be included in the scope of the measured size.

6.3 Identification of the FUR

The Functional User Requirements identified to be within the scope of the FSM shall be identified and used as the exclusive viewpoint from which the functional size of the software is to be measured.

Note Informative Annex A contains guidance about how to extract FUR from various sources.

All users of the FUR within the scope of FSM shall be identified.

6.4 Identification of software layers

6.4.1 Identify software layers

Software may have components of its functionality that exist at different levels of functional abstraction in the software's operating environment, where there is a hierarchical relationship between the levels of abstraction based on the functional exchanges among them. If required for the purpose of the measurement exercise, each such level shall be designated as a distinct layer.

NOTE 1 Functional User Requirements may state explicitly, may imply, or the measurement analyst may infer, that the FUR apply to software in different layers or peer items whose size must be measured separately. Alternatively, the measurement analyst may be faced with sizing existing software which appears to be in different layers or peer items. In both cases, guidance is needed (see 6.4.2) to help decide if FUR or the software comprise one or more layers or peer items.

NOTE 2 Layer identification is an iterative activity. The exact identification of layers will be refined as the measurement activity progresses.

NOTE 3 Software within a layer may be developed and need to be sized as separate 'pieces'.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

NOTE 4 Data movements between separate pieces of software within the same layer are known as 'peer-to-peer' communication.

Examples are provided in Informative Annex B.

6.4.2 Characteristics of software layers

Layers identified within the scope of the FSM shall have the following characteristics:

a) Software in each layer shall deliver functionality to its users;

b) Software in a subordinate layer shall provide functional services to software in a layer using its services;

c) Software that shares data with other software shall not be considered to be in different layers if they identically interpret the data attributes that they share.

NOTE 1 Software in a subordinate layer could perform without assistance from software in a layer using its services.

NOTE 2 Software in one layer might not perform properly if software in a subordinate layer on which it depends is not performing properly.

NOTE 3 Software in one layer does not necessarily use all the functionality supplied by software in a subordinate layer.

NOTE 4 In a hierarchy of layers, software in any one layer can be a subordinate to software in a higher layer for which it provides services.

NOTE 5 Software in one layer and in a subordinate layer can physically share and exchange data. However, the software in each layer will interpret the data attributes differently and/or group them in different data groups.

NOTE 6 The concept of software layers is a tool to help differentiate Functional User Requirements allocated at different levels of functional abstraction.

NOTE 7 There are many software architecture models in use. The layered model is used here to provide a functional view of the software. Other models could be used if they provide a functional view of the software, fully or partly.

NOTE 8 The concept of layers presented in this International Standard is different from the concept of "layered architecture". Although there are elements common to both concepts, the COSMIC-FFP layers are meant as a tool to help the practitioner in the identification of the scope and the boundaries. If a specific architectural paradigm is used within an organisation, then an equivalence should be established between specific architectural elements in that paradigm and the concept of layers as defined in this International Standard, and the resulting layers should be used.

NOTE 9 Functional service packaged software, such as database management systems, graphical user interfaces, operating systems or device drivers, are generally considered as distinct layers.

6.5 Identification of software boundaries

The requirements for identifying boundaries are:

a) The boundary of each piece of software within each layer identified to be in the scope of the FSM shall be identified;

b) Once the boundaries have been identified, each identified FUR within the scope of the FSM shall be allocated to a piece of software.

NOTE 1 When identifying a COSMIC-FFP boundary, the following guidelines should be applied:

c) Start by identifying Triggering events, then identify the functional processes enabled by those events. The boundary lies between the Triggering events and those functional processes;

d) Use the concept of layers to assist in the identification of the boundary.

© ISO/IEC 2003 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

NOTE 2 Boundary identification is an iterative activity. The exact boundary of any piece of software being measured will be refined as the measurement activity progresses.

NOTE 3 In any given layer, the piece of software to be measured can be clearly distinguished from the surrounding layers by a boundary. An implicit boundary thus exists between identified layers. All users of a piece of software lie outside the boundary of this piece of software.

6.6 Identification of functional processes

Each functional process (-type) identified in the scope of the FSM shall be derived from at least one identifiable Functional User Requirement, and

a) be initiated by a triggering event,

b) comprise at least two data movements - an Entry plus either an Exit or a Write,

NOTE According to this characteristic and using 1 Cfsu as the unit of measurement, the smallest theoretical functional size for a piece of software is 2 Cfsu.

c) belong to one, and only one layer, and

d) complete when a point of asynchronous timing is required to be reached according to its FUR.

NOTE A point of asynchronous timing is reached when the final (terminating) data movement in a sequence of data movements is not synchronized with any other data movement.

6.7 Identification of data groups

Each data group identified in the scope of the FSM shall:

a) be unique and distinguishable through its unique collection of data attributes, and

b) be directly related to one object of interest described in the software's Functional User Requirements.

6.8 Identification of data movements

Each functional process identified in 6.6 shall be partitioned into its component data movements.

Each data movement shall:

a) be either an Entry, Exit, Read or Write,

b) be counted only once within the functional process where it is identified, and

c) not involve another data movement-type.

NOTE A given data movement type may occur many times within a given functional process, e.g. as in a loop, but for sizing purposes, data movement types are identified, not their occurrences.

6.9 Classification of data movements

6.9.1 Entry

An Entry shall:

a) receive data attributes from a single data group which originates from the user side of the boundary,

NOTE If the data attributes entering a functional process originated from more than one data group, then identify one Entry for each data group.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

b) include all required formatting and presentation manipulations along with all associated validations of the entered data attributes, to the extent that these manipulations do not involve another type of data movement.

NOTE An Entry includes all manipulations except the Read(s) that might be required to validate some entered codes or to obtain some associated descriptions.

c) include any 'request to receive the Entry' functionality.

6.9.2 Exit

An Exit shall:

a) send data attributes from a single data group to the user side of the boundary,

NOTE If data belonging to more than one data group in a functional process are sent outside the boundary, identify one Exit for each referenced data group.

b) include all required data formatting and presentation manipulations, including processing required to send the data attributes to the user, to the extent that these manipulations do not involve another type of data movement.

c) include any 'request to Read' functionality.

6.9.3 Read

A Read shall:

a) retrieve data attributes from a single data group from persistent storage,

NOTE If more than one data group is retrieved from persistent storage in a functional process, identify one Read for each data group.

b) include all logical processing and/or mathematical computation needed to read the data, to the extent that these manipulations do not involve another type of data movement.

NOTE During a functional process, the Read or Write of a data group can only be performed on the data describing an Object of Interest to the User. Constants or variables which are internal to the functional process, or intermediate results in a calculation, or data stored by a functional process resulting only from the implementation, rather than the FUR, are not data groups and are not counted.

6.9.4 Write

A Write shall:

a) move data attributes from a single data group to persistent storage,

NOTE If more than one data group is written to persistence storage in a functional process, identify one Write for each data group.

b) include all logical processing and/or mathematical computation to create the data attributes to be written, to the extent that these manipulations do not involve another type of data movement.

A requirement to delete a data group from persistent storage shall be a single Write data movement.

NOTE During a functional process, the storage of a data group that does not persist when the functional process is complete is not a Write; examples are variables which are internal to the functional process or intermediate results in a calculation.

© ISO/IEC 2003 · All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.10 Calculation of the Functional Size

6.10.1 Assignment of a size unit

A unit of measurement, 1 Cfsu, shall be assigned to each data movement identified, that is to each instance of a data movement (Entry, Exit, Read or Write) identified.

6.10.2 Aggregation of functional size for a functional process

The results of 6.10.1, as applied to all identified data movements within the identified functional process, shall be aggregated into a single functional size value for that functional process by:

a) multiplying the number of data movements within each data movement type by its unit size,

b) obtaining the size of the functional process by totaling the sizes of each of its data movements,

Thus, the Functional Size of a given Functional Process (FP Size) is calculated in Cfsu using the formula:

$$\text{FP Size} = \Sigma\,(\text{Ne} * \text{Eus}) + \Sigma\,(\text{Nx} * \text{Xus}) + \Sigma\,(\text{Nr} * \text{Rus})\text{i} + \Sigma\,(\text{Nw} * \text{Wus})$$

Where:

Ne = number of Entries for the FP, Eus = Entry Unit Size

Nx = number of Exits for the FP, Xus = Exit Unit Size

Nr = number of Reads for the FP, Rus = Read Unit Size

Nw = number of Writes for the FP, Wus = Write Unit Size

NOTE 1 The Unit Size is 1 for all data movement types.

NOTE 2 There is no upper limit to the functional size of a functional process.

6.10.3 Aggregation of functional size for the identified FURs for each piece of software to be measured

The size of each piece of software to be measured within a layer shall be obtained by aggregating the size of the functional processes within the identified FURs for each piece.

NOTE 1 Within each identified layer, the aggregation function is fully scalable. Therefore a sub-total can be generated for individual functional processes, individual software pieces or for the whole layer, depending on the purpose and scope of the measurement exercise.

NOTE 2 Aggregating the measurement results by type of data movements might be useful for analysing the contribution of each type to the total size of a layer and might thus help characterise the functional nature of the measured layer.

NOTE 3 In a context where functional size is to be used as a variable in a model, to estimate effort for instance, and the software to be sized has more than one layer, aggregation will typically be performed up to the layer level since layers are not necessarily implemented with the same technology.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6.10.4 Calculation of Functional Size of changes to the FURs

Within each identified layer, the functional size of changes to the Functional User Requirements within each piece of software within the scope of the FSM shall be calculated by aggregating the sizes of the corresponding impacted data movements according to the following formula:

$$Size_{Cfus}\ (Change) = \sum size(added\ data\ movement_i) + \sum size(changed\ data\ movement_i) + \sum size(deleted\ data\ movement_i)$$

NOTE A data movement is considered to be changed if any of the attributes of the data group are changed, or if any changes are needed to the data manipulation associated with the data movement.

EXAMPLE A requested change is: add one new Functional Process of size 6 Cfsu, and in another Functional Process add one Data Movement, make changes to three other Data Movements and to delete two Data Movements. The total size of the requested change is 6 + 1 + 3 + 2 = 12 Cfsu.

7 Local customizations

The COSMIC-FFP Functional Size Measurement Method is currently not designed to provide a way of measuring the size of certain types of Functional User Requirements, notably complex mathematical algorithms or complex sequences of rules as found in expert systems (see 1.3). However, within the local environment of an organisation using the COSMIC-FFP Functional Size Measurement Method, it may be possible to measure these FUR in a way which is meaningful as a local standard.

The COSMIC-FFP Functional Size Measurement Method provides for local customizations. In any functional process where there is abnormally complex data manipulation, the measurement analyst may assign his or her own locally-determined functional size units in this situation.

When a local customization is used, the measurement results shall be reported according to 8.2.

EXAMPLE A local customization standard could be: In our organisation, we assign one Local Functional Size Unit for mathematical algorithms such as (list of locally meaningful and well-understood examples…). We assign two Local Functional Size Units for (another list of examples), etc.

8 Measurement Reporting

8.1 Labelling

A COSMIC-FFP measurement result on the FUR for a piece of software shall be labelled according to the following convention:

Cfsu (COSMIC-FFP Ver. ISO/IEC 19761:2002)

8.2 Customization

A COSMIC-FFP measurement result on the FUR of a piece of software using local customizations shall be recorded as:

"x Cfsu(COSMIC-FFP Ver. ISO/IEC 19761: 2002 + z Local FSU", where:

"x" represents the numerical value obtained by aggregating all individual measurement results for the given piece of software according to the International Standard COSMIC-FFP method, version v,y;

"z" represents the numerical value obtained by aggregating all individual measurement results obtained from a local customization to the COSMIC-FFP method for the given piece of software.

--`,,`,,`-`-`,,`,,`,`,,`---

© ISO/IEC 2003 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

8.3 Documentation of the measurement results

The documentation of the COSMIC-FFP measurement results shall include the following information:

a) Identification of each piece of software in the scope of the FSM (name, version identification or configuration identification);

b) A description of the measurement purpose and scope (the 'viewpoint');

c) A description of the relationship of each piece of software within the scope of the FSM with its users and the position of the boundaries, both peer-to-peer and between layers;

d) The functional size value of each piece of software within the scope of the FSM, calculated according to 6.10 and reported according to 8.1 or 8.2.

In addition, the documentation of the COSMIC-FFP measurement results should include the following information for each piece of software within the scope of the FSM:

e) A list of the functional processes identified and their associated data movement types;

f) A list of the data groups identified;

g) The total number of functional processes identified;

h) The total number of data groups identified;

i) The total functional size of the Entries;

j) The total functional size of the Exits;

k) The total functional size of the Reads;

l) The total functional size of the Writes.

NOTE Documentation is required for each piece of software measured within each layer.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex A
(informative)

Extraction of Functional User Requirements

There are many reasons to measure the functional size of software. In a particular context, it might be necessary to measure the functional size of software prior to its development. In a different context, it will be useful to measure the functional size of software *a posteriori*, that is, some time after it has been put into production.

The reasons why a measurement is taken has an impact, albeit at times subtle, on what is being measured, either the object of interest itself, or its representation, without affecting the unit of measurement or the measurement principles.

Measuring the functional size of software prior to its development is based on the software "plans"; a collection of artifacts produced prior to development. The required dimensions (Base Functional Components) are extracted from the artifacts using appropriate conventions (from the perspective of the Functional User Requirements, excluding technical and quality aspects) and the size is calculated according to a specific measurement function.

Likewise, measuring the functional size of software after it has been put into production entails a somewhat different measurement procedure when the required dimensions are extracted from the various artifacts. Although the nature of these artifacts differs, the dimensions, the unit of measurement and the measurement principles remain the same. There are many aspects of software. From the perspective of the COSMIC-FFP Functional Size Measurement Method, the aspect of interest is the functionality it delivers to its users. The functionality delivered by software to its users is described through the Functional User Requirements (FUR). In practice, FUR sometimes exist in the form of a specific document (requirements specifications, for instance), but often they have to be derived from other software engineering artifacts. As illustrated in Figure A.1, FUR can be derived from software engineering artifacts that are produced before the software exists (typically from architecture and design artifacts). Thus, the functional size of software can be measured prior to its implementation on a computer system.

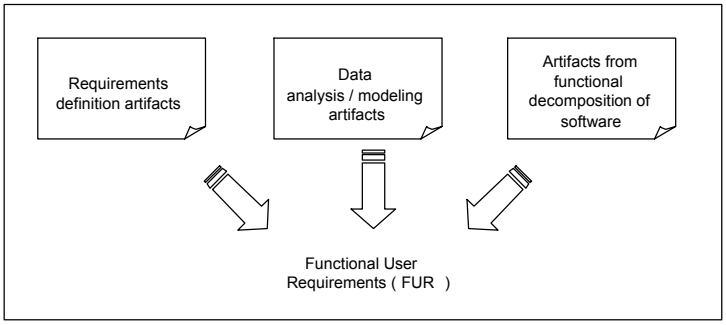

Figure A.1 — Extracting FUR at pre-implementation of software

In other circumstances, software might be used without there being any, or with only a few, architecture or design artifacts available, and the Functional User Requirements might not be documented (legacy software, for instance). In such circumstances, it is still possible to derive the software Functional User Requirements from the artifacts installed on the computer system, as illustrated in Figure A.2.

The effort required to extract the FUR from different types of software engineering artifacts will obviously vary. The nature of the FUR remains constant, though, since no matter what type of software engineering artifacts are used to extract them, they always convey a description of the functionality delivered by the software to its users.

© ISO/IEC 2003 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

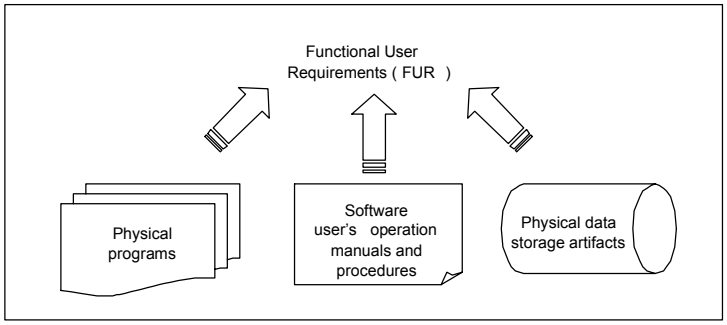

Figure A.2 — Extracting FUR at post-implementation of software

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex B
(informative)

Allocation of Functional User Requirements (FUR)

The following are two examples of allocation of software functional requirements: typical business or MIS application software and “multi-piece” software.

EXAMPLE 1 Typical MIS application software

In this example, Functional User Requirements are typically allocated to the software application “layer” (“Application X” in Figure B.1). The other pieces of software in the environment are generally required and used without modification. The example in Figure B.1 illustrates this. Since

a) all Functional User Requirements are allocated to a single piece of software, and

b) this piece of software can be easily distinguished from other pieces of software within the environment through a well-defined functional interface,

it is a relatively simple matter to identify the functionality offered by Application X and, thus, to measure the functionality lying inside the Application X boundary. In this example and for the purpose of Functional Size Measurement, an abstraction could be made by ignoring all software in the environment except Application X, in which case the Application X boundary (solid line around the gray box in Figure B.1 would be undistinguishable from the software environment’s boundary (dashed line delimiting software from hardware in Figure B.1). In this abstraction, the user(s) could be the human user(s) of the application.

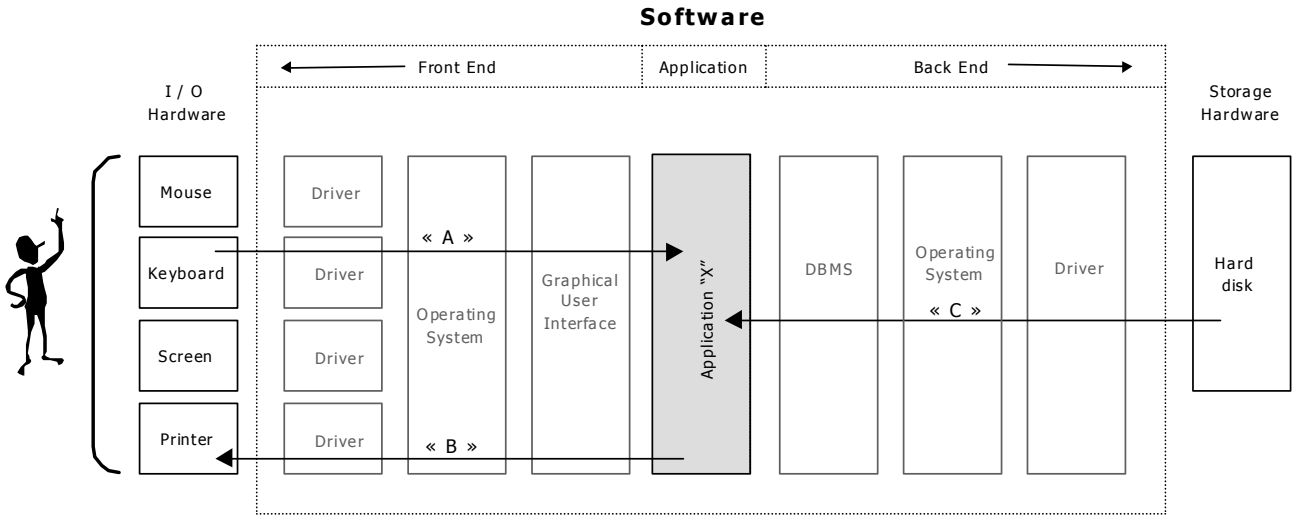

Figure B.1 — Example of requirements allocation for MIS software

EXAMPLE 2 Multi-piece software

In this example, one part of the Functional User Requirements is allocated to an application (“Application A” in Figure B.2) and another part is allocated to a custom graphical user interface (“Graphical User Interface” in Figure B.2) and the purpose is to measure the size of the two components separately. The functional interface between Application A and the graphical user interface is also structured and well defined in the Functional User Requirements. The other pieces of software in the environment are used without modification and the required software simply makes use of these pieces as needed. The example in Figure B.2 illustrates this.

Copyright International Organization for Standardization · All rights reserved
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

There is a dependency between Application A and the graphical user interface: Application A relies on the graphical user interface by invoking the functionality provided by the latter in order to deliver its "own" functionality. Because of this close relationship, the functionality provided by the graphical user interface is not separately visible to the external human user. The latter sees the functionality provided by the graphical user interface and Application A without distinction. But in order to measure the functional size of the total requirements, it is necessary to distinguish between these two pieces. For this purpose, the COSMIC-FFP Functional Size Measurement Method will place each piece on a different layer, and each piece will have its own boundary. In this example, given that the interface between the two pieces is structured and well defined, the distinction should not cause great difficulty in practice for the measurement analyst. Finally, it is to be noted that each piece will be considered a "user" of the other piece for the purpose of measuring their functional size.

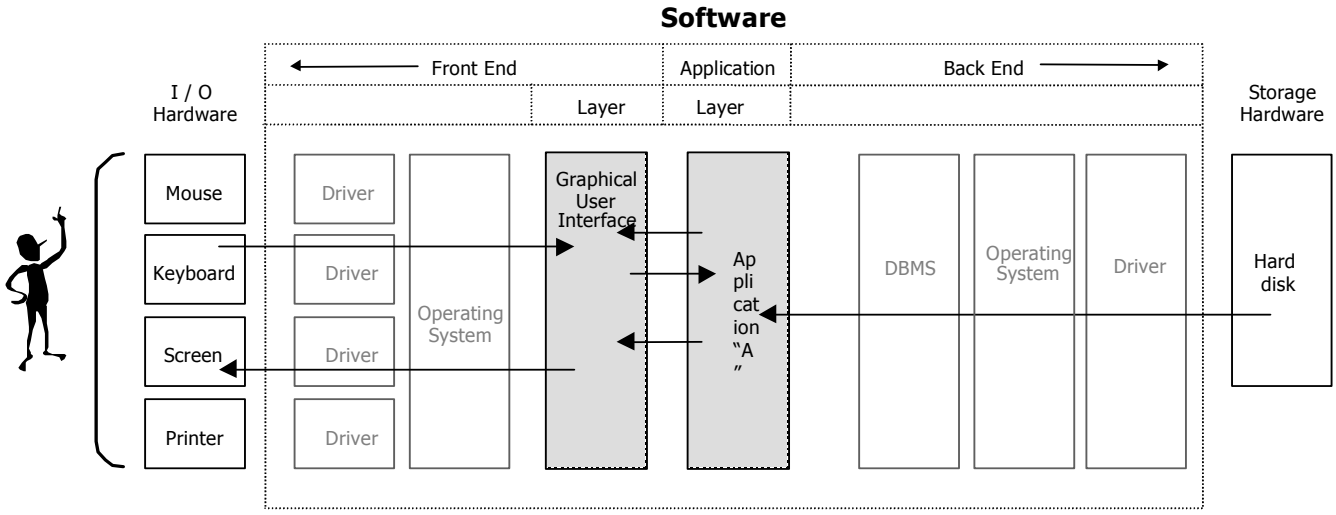

Figure B.2 — Example of requirements allocation for "multi-piece" software

In software engineering practice, Functional User Requirements are not necessarily allocated to a single piece of software. In cases where a software architecture exists early in the construction process, such as the layered architecture illustrated in Figure B.1 for instance, the allocation of Functional User Requirements to specific pieces in this architecture has an impact on the functional size of each piece.

For practitioners interested only in the global size of these requirements, it is not necessary to consider how the requirements are allocated (Example 1). For practitioners who are interested in the functional size of portions of the requirements (Example 2), this allocation scheme becomes relevant to the identification of the software boundary.

© ISO/IEC 2003 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Bibliography

[1] Abran, A.; Desharnais, J.-M.; Oligny, S.; St-Pierre, D.; Symons, C., *COSMIC FFP — Measurement Manual version 2.1*, Montréal (Canada), May, 2001. http://www.lrgl.uqam.ca/cosmic-ffp

[2] ISO, *International Vocabulary of Basic and General Terms in Metrology*, 1993

[3] ISO/IEC 15939, *Software engineering — Software measurement process*

--`,,`,,`-`-`,,`,,`,`,,`---

© ISO/IEC 2003 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

ICS 35.080

Price based on 17 pages

© ISO/IEC 2003 – All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale