DRAFT INTERNATIONAL STANDARD ISO/IEC 20926

Attributed to ISO/IEC JTC 1 by the Central Secretariat (see page iii)

Voting begins on	Voting terminates on
2008-06-05	**2008-12-05**

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОММИСИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

PUBLICLY AVAILABLE SPECIFICATION PROCEDURE

Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009

[Revision of first edition (ISO/IEC 20926:2003)]

Ingénierie du logiciel et des systèmes — Mesurage du logiciel — Méthode IFPUG 2009 de mesurage de la taille fonctionnelle

ICS 35.080

In accordance with the provisions of Council Resolution 21/1986 this DIS is circulated in the English language only.

Conformément aux dispositions de la Résolution du Conseil 21/1986, ce DIS est distribué en version anglaise seulement.

This Publicly Available Specification (PAS) is being submitted for Fast-track processing in accordance with the provisions of ISO/IEC JTC 1 Directives.

THIS DOCUMENT IS A DRAFT CIRCULATED FOR COMMENT AND APPROVAL. IT IS THEREFORE SUBJECT TO CHANGE AND MAY NOT BE REFERRED TO AS AN INTERNATIONAL STANDARD UNTIL PUBLISHED AS SUCH.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

RECIPIENTS OF THIS DRAFT ARE INVITED TO SUBMIT, WITH THEIR COMMENTS, NOTIFICATION OF ANY RELEVANT PATENT RIGHTS OF WHICH THEY ARE AWARE AND TO PROVIDE SUPPORTING DOCUMENTATION.

© International Organization for Standardization, 2008
© International Electrotechnical Commission, 2008

Copyright International Organization for Standardization
Provided 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.

Copyright notice

This ISO document is a Draft International Standard and is copyright-protected by ISO. Except as permitted under the applicable laws of the user's country, neither this ISO draft nor any extract from it may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission being secured.

Requests for permission to reproduce should be addressed to 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

Reproduction may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

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

© ISO/IEC 2008 — All rights reserved

Not for Resale

NOTE FROM ITTF

The ballot on the transposition of a PAS into an international standard follows the JTC 1 PAS procedures contained in the JTC 1 Directives Section 14.

In the PAS process, the JTC 1 secretariat will inform JTC 1 National Bodies and Liaison Organisations, and those organisations authorised to be PAS submitters, of the initiation of the PAS ballot, the result of the ballot, and the identity of the JTC 1 committee which will be responsible for any future work.

For ballot, JTC 1 National Bodies and the PAS submitter will receive both the PAS to be transposed and the accompanying Explanatory Report. During the ballot JTC 1 members may propose changes to the PAS. These can be resolved with the PAS Submitter after completion of the ballot.

The period for combined DIS voting shall be six months. In order to be accepted the DIS must be supported by 75 % of the votes cast (abstention is not counted as a vote) and by two-thirds of the P-members voting of JTC 1.

In the case of a failure of the ballot, JTC 1 will make known to the Submitter the reasons which have led to the negative result. Based on this information, the Submitter may choose to re-submit the specification after modification.

Once the Draft International Standard has been approved by JTC 1 as an IS, it will be published.

© ISO/IEC 2008 – All rights reserved

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

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

Contents

Foreword iv

Introduction v

1 Scope 1
1.1 Purpose 1
1.2 Applicability 1
1.3 Audience 1

2 Terms, definitions, units and abbreviated terms 1

3 Measurement Process 7
3.1 Overview 7
3.2 Gather the available documentation 8
3.3 Determine the counting scope and application boundaries 8
3.4 Measure data functions 9
3.4.1 Overview 9
3.4.2 Identify and group all logical data into data functions 9
3.4.3 Classify each data function as either an ILF or an EIF 10
3.4.4 Count DETs for each data function 10
3.4.5 Count RETs for each data function 10
3.4.6 Determine the functional complexity for each data function 11
3.4.7 Determine the functional size for each data function 11
3.5 Measure transactional functions 11
3.5.1 Overview 11
3.5.2 Identify elementary processes 12
3.5.3 Classify each elementary process as a transactional function 14
3.5.4 Count FTRs for each transactional function 15
3.5.5 Count DETs for each transactional function 15
3.5.6 Determine the functional complexity for each transactional function 16
3.5.7 Determine the functional size of each transactional function 17
3.6 Measure conversion functionality 17
3.7 Measure enhancement functionality 17
3.8 Calculate functional size 18
3.9 Document the function point count 19
3.10 Report the result of the function point count 19

Annex A 21

DRAFT 2008

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

Copyright International Organization for Standardization ll rights reserved
Provided 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.

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. 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 International Standard may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 20926 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information Technology*, Subcommittee SC 7, *Software and Systems Engineering*.

This second edition cancels and replaces the first edition (20926: 2004) in its entirety which has been technically revised.

DRAFT 2008

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

© ISO/IEC 2004 – All rights reserved

Introduction

The use of function points, as a measure of the functional size of software, has grown since the mid 1970s from a few interested organizations to an impressive list of organizations worldwide. Allan Albrecht was the first to publicly release a method for functionally sizing software called function point analysis. With the growth in the use of function points, there has been wider application and use of the measure. Since its formation in 1986 the International Function Point Users Group (IFPUG) has continuously enhanced the original Albrecht method for functionally sizing software. This International Standard is the latest release in the continually improving IFPUG method that promotes the consistent interpretation of functional size measurement in conformance with ISO/IEC 14143-1:2007. The IFPUG functional size measurement method is known as function point analysis and its units of functional size are called Function Points.

Organizations can apply this International Standard to measure the size of a software product to:

— support quality and productivity analysis;

— estimate cost and resources required for software development, enhancement and maintenance;

— provide a normalization factor for software comparison;

— determine the size of a purchased application package by functionally sizing all the functions included in the package;

— assist users in determining the benefit of an application package to their organization by functionally sizing functions that specifically match their requirements.

Function point analysis measures software by quantifying the tasks and services (i.e., functionality) that the software provides to the user based primarily on logical design. The objectives of function point analysis are to measure:

— functionality implemented in software, that the user requests and receives;

— functionality impacted by software development, enhancement and maintenance independently of technology used for implementation.

The process of function point analysis is:

— simple enough to minimize the overhead of the measurement process;

— a consistent measure among various projects and organizations.

This document is one component in the IFPUG publications. It is recommended that this document be read in conjunction with the other IFPUG publications. These provide guidance to application of the rules described within this International Standard and background information to aid in understanding the use and applicability of the resulting functional size. Supporting IFPUG publications include the following:

— The current version of the IFPUG Counting Practices Manual, which incorporates this International Standard supplemented with counting practices and examples that support its implementation;

— “Framework for Functional Sizing”, 2005, which discusses the contribution of both functional size and non-functional size to the overall software product size; the IFPUG FPA method is a method for measuring the functional size;

— IFPUG website at www.ifpug.org.

DRAFT 2008

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

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

Software and systems engineering – Software measurement – IFPUG functional size measurement method 2009

1 Scope

1.1 Purpose

This International Standard specifies the set of definitions, rules and steps for applying the IFPUG functional size measurement (FSM) method.

1.2 Applicability

This International Standard has been demonstrated to apply to all functional domains. IFPUG continues to publish white papers providing guidelines for use in evolving environments and domains.

IFPUG function point analysts have identified different delivery rates (hours to deliver a function point) related to building applications in different functional domains calibrated for varying project sizes and software complexities.

1.3 Audience

This International Standard can be applied by anyone requiring a measurement of functional size. Persons experienced with the method will find this document to be a useful reference. In order to effectively apply this International Standard, persons should be formally trained in the method using IFPUG Certified course materials. It is also advisable to use this standard in conjunction with the other listed IFPUG publications.

2 Terms, definitions, units and abbreviated terms

2.1
adaptive maintenance
software maintenance performed to make a computer program usable in a changed environment

NOTE Contrast with: corrective maintenance; perfective maintenance.

[IEEE Std. 610.12-1990]

2.2
application
cohesive collection of automated procedures and data supporting a business objective; it consists of one or more components, modules, or subsystems

EXAMPLES accounts payable, accounts receivable, payroll, procurement, shop production, assembly line control, air search radar, target tracking, weapons firing, flight line scheduling and passenger reservations.

2.3
application functional size
a measure of the functionality that an application provides to the user, determined by the application function point count

DRAFT 2008

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

2.4
application function point count
activity of applying this International Standard to measure the functional size of an application

2.5
arranging
the activity of sequencing attributes in a transactional function

2.6
associative entity type
an entity type that contains attributes which further describe a many-to-many relationship between two other entity types

2.7
attributive entity type
an entity type that further describes one or more attributes of another entity type

2.8
boundary
a conceptual interface between the software under study and its users

[ISO/IEC 14143-1:2007, definition 2.3]

2.9
consistent state
the point at which processing has been fully executed, the functional requirement has been satisfied and there is nothing more to be done

EXAMPLE 1 The functional requirement is to print a check and mark the appropriate account as paid. If only part of the functional requirement is completed (e.g., only printing the check or only marking it as paid), the application is not in a consistent state. The printing of a check without marking the account as paid causes an inconsistency in the application as does marking it as paid without printing it.

EXAMPLE 2 The functional requirement is to have a batch process that accepts an input file to update a data store, produce a production control report and return an error report back to the sending application. The process is not in a consistent state unless all of these parts are completed.

EXAMPLE 3 The functional requirement is to transfer an employee to a new job and validate his/her security clearance level. To complete this, a real-time request is sent to the security application (which maintains government security clearances and not application security) and a response received before the transfer can be completed. All steps are needed to create a consistent state. The interaction with the security application is not an independent step or action. It does not happen in and of itself, and the transaction to transfer an employee is not in a consistent state without it.

2.10
control information
data that influences an elementary process by specifying what, when or how data is to be processed

2.11
conversion functionality
transactional or data functions provided to convert data and/or provide other user specified conversion requirements

NOTE Conversion functionality exists only during the development or enhancement of an application.

2.12
corrective maintenance
maintenance performed to correct faults in hardware or software

NOTE Contrast with: adaptive maintenance; perfective maintenance.

DRAFT 2008

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale
© ISO/IEC 2004 – All rights reserved

[IEEE Std. 610.12-1990]

2.13
counting scope
the set of functional requirements to be included in the function point count

2.14
Data Element Type
DET
a unique, user recognizable, non-repeated attribute

2.15
data function
functionality provided to the user to meet internal or external data storage requirements

NOTE A data function is either an internal logical file or an external interface file.

2.16
derived data
data created as a result of processing that involves steps, other than or in addition to, direct retrieval and validation of information from data functions

2.17
development project
a project to develop and deliver the first release of a software application

2.18
development project functional size
a measure of the functionality provided to the users with the first release of the software, as measured by the development project function point count

NOTE The functional size of a development project may include the size of conversion functionality.

2.19
development project function point count
the activity of applying this International Standard to measure the functional size of a development project

2.20
Elementary Process
EP
the smallest unit of activity that is meaningful to the user

2.21
enhancement project
a project to develop and deliver adaptive maintenance

NOTE An enhancement project may also develop and deliver corrective and perfective maintenance, but these do not contribute to the enhancement project functional size.

2.22
enhancement project functional size
a measure of the functionality added, changed or deleted at the completion of an enhancement project, as measured by the enhancement project function point count

NOTE The functional size of an enhancement project may include the size of conversion functionality.

2.23
enhancement project function point count
the activity of applying this International Standard to measure of the functional size of an enhancement project

DRAFT 2008

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

2.24
entity dependent
an entity that is not meaningful or is not significant to the business, in and of itself without the presence of other entities such that

- an occurrence of entity X must be linked to an occurrence of entity Y, and
- the deletion of an occurrence of entity Y results in the deletion of all related occurrences of entity X.

2.25
entity independent
an entity that is meaningful or significant to the business, in and of itself without the presence of other entities

2.26
External Input
EI
an elementary process that processes data or control information sent from outside the boundary

2.27
External Inquiry
EQ
an elementary process that sends data or control information outside the boundary

2.28
External Interface File
EIF
a user recognizable group of logically related data or control information, which is referenced by the application being measured, but which is maintained within the boundary of another application

2.29
External Output
EO
an elementary process that sends data or control information outside the boundary and includes additional processing logic beyond that of an External Inquiry

2.30
File Type Referenced
FTR
a data function read and/or maintained by a transactional function

2.31
functional complexity
the specific complexity rating assigned to a function using the rules as defined within this standard

2.32
functional size
a size of the software derived by quantifying the functional requirements

[14143-1:2007, definition 2.6]

2.33
functional user requirement
functional requirement
a sub-set of the user requirements; requirements that describe what the software shall do, in terms of tasks and services

NOTE Functional user requirements include but are not limited to:

- data transfer (for example: input customer data, send control signal);

DRAFT 2008

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

- data transformation (for example: calculate bank interest, derive average temperature);
- data storage (for example: store customer order, record ambient temperature over time);
- data retrieval (for example: list current employees, retrieve aircraft position).

Examples of user requirements that are not functional user requirements include but are not limited to:

- quality constraints (for example: usability, reliability, efficiency and portability);
- organizational constraints (for example: locations for operation, target hardware and compliance to standards);
- environmental constraints (for example: interoperability, security, privacy and safety);
- implementation constraints (for example: development language, delivery schedule).

[14143-1:2007, definition 2.8]

2.34
Function Point
FP
the unit of measure for functional size as defined within this International Standard

2.35
Function Point Analysis
FPA
the method for measuring functional size as defined within this International Standard

2.36
function point count
the activity of applying the rules within this International Standard to measure the functional size of an application or project

NOTE There are three types of function point counts: application, development project and enhancement project.

2.37
function type
the five base functional components identified in this International Standard

NOTE The five function types are: External Input, External Output, External Inquiry, Internal Logical File and External Interface File.

2.38
Internal Logical File
ILF
a user recognizable group of logically related data or control information maintained within the boundary of the application being measured

2.39
maintain
the ability to add, change or delete data through an elementary process

2.40
meaningful
user recognizable and satisfies a functional requirement

2.41
perfective maintenance
software maintenance performed to improve the performance, maintainability or other attributes of a computer program

DRAFT 2008

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

NOTE Contrast with: adaptive maintenance; corrective maintenance.

[IEEE Std. 610.12-1990]

2.42
primary intent
first in importance

2.43
processing logic
any of the requirements specifically requested by the user to complete an elementary process such as validations, algorithms or calculations and reading or maintaining a data function

2.44
purpose of the count
the reason for performing the function point count

NOTE The purpose of the count determines the counting scope.

EXAMPLE 1 The purpose of the count could be to determine the size of a particular software release.

EXAMPLE 2 The purpose of the count could be to determine the size of an application as part of the organization's effort to determine the size of its software portfolio.

2.45
Record Element Type
RET
a user recognizable sub-group of data element types within a data function

2.46
self-contained
no prior or subsequent processing steps are needed to initiate or complete the functional requirement(s)

EXAMPLE The functional requirement states that an employee is to be both added and updated. There may be multiple parts that make up the complete employee information. This may be represented by separate physical screens, windows or tabs such as

- Employee identification,
- Employee location,
- Dependent information,
- Salary information, and
- Education.

To add an employee, one or more of the tabs must be completed, depending on the business rules. The add process is not self-contained until all mandatory information has been entered.

To update an employee, one or more of the tabs can be updated at any given time, but they are all process steps that meet the functional requirement of updating an employee. Adding, changing or deleting information on any individual tab is not a separate elementary process. It is a process step involved in updating an employee. Even though additional information is possible to be entered into the employee record, all of the information together is considered to be a part of the single elementary process: update employee.

Add Employee and Update Employee would each be a self-contained process.

2.47
sorting
the activity of sequencing of rows or records in a transactional function

DRAFT 2008

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

2.48
transactional function
an elementary process that provides functionality to the user to process data

NOTE A transactional function is an External Input, External Output or External Inquiry.

2.49
user
any person or thing that communicates or interacts with the software at any time

NOTE Examples of 'thing' include, but are not limited to, software applications, animals, sensors, or other hardware.

[ISO/IEC 14143-1:2007, definition 2.11]

2.50
user recognizable
requirements for processes and/or data that are agreed upon, and understood by, both the user(s) and software developer(s)

2.51
user view
functional requirements as perceived by the user

NOTE Developers translate the user view into software in order to provide a solution.

3 Measurement Process

3.1 Overview

To conduct a function point count

a) gather the available documentation in accordance with 3.2,

b) determine the counting scope and application boundaries in accordance with 3.3,

c) measure data functions in accordance with 3.4,

d) measure transactional functions in accordance with 3.5,

e) for a development or enhancement project function point count, measure conversion functionality in accordance with 3.6,

f) for an enhancement project function point count, measure added, changed and deleted functionality in accordance with 3.7,

g) calculate the functional size in accordance with 3.8,

h) document the Function Point Count in accordance with 3.9, and

i) report the result of the Function Point Count in accordance with 3.10.

NOTE Figure 1 provides a graphical overview of the function point counting process.

DRAFT 2008

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

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

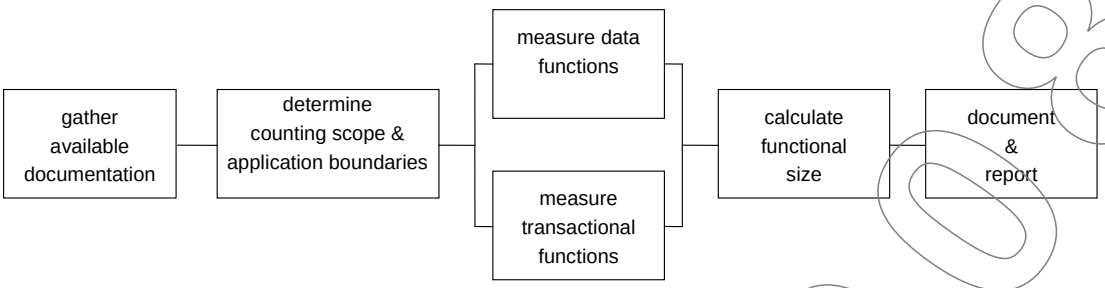


Figure 1 — Graphical overview of the function point counting process

3.2 Gather the available documentation

Documentation to support a function point count must describe the functionality delivered by the software or the functionality that is impacted by the software project that is being measured.

Obtain sufficient documentation to conduct the function point count or access to subject matter experts who are able to provide additional information to address any gaps in the documentation.

NOTE Suitable documentation may include requirements, data/object models, class diagrams, data flow diagrams, use cases, procedural descriptions, report layouts, screen layouts, user manuals and other software development artifacts.

3.3 Determine the counting scope and application boundaries

To determine the counting scope and application boundaries

a) Identify the purpose of the count,

NOTE A function point count is conducted to provide an answer to a business question, and it is the business question that determines the purpose.

b) Identify the type of count, based on the purpose, as one of:

   1) a development project function point count;

   2) an application function point count;

   3) an enhancement project function point count,

c) Determine the counting scope, based on the purpose and type of count,

d) Determine the boundary of the application(s) within the counting scope, based on the user view, not on technical considerations

EXAMPLE If the purpose is to estimate the cost of an enhancement to the HR and Benefits applications, then:

- the type of count will be an enhancement project count;
- the scope will be the transactional and data functions added, changed or deleted for both the HR and Benefits applications as well as any conversion requirements for each application;
- the user view is that HR and Benefits are different functional areas, and therefore they are separate applications;
- a boundary exists between the HR and Benefits applications as well as between each application and user.

DRAFT 2008

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

e) The user requirements may include a mixture of both functional and non-functional requirements; identify which requirements are functional, and

f) Exclude the non-functional requirements.

3.4 Measure data functions

3 4.1 Overview

Data functionality satisfies the functional requirements to store and/or reference data. All data functionality within the counting scope needs to be evaluated to identify each logical group of data.

To measure data functions

a) identify and group all logical data into data functions in accordance with 3.4.2,

b) classify each data function as either an ILF or an EIF in accordance with 3.4.3,

c) count the DETs for each data function in accordance with 3.4.4,

d) count the RETs for each data function in accordance with 3.4.5,

e) determine the functional complexity for each data function in accordance with 3.4.6, and

f) determine the functional size for each data function in accordance with 3.4.7.

3 4.2 Identify and group all logical data into data functions

NOTE 1 Data functions are most easily identified using a logical data model; however, this does not preclude the use of the measurement process in environments where alternative data or object modeling techniques are employed. Data modeling terminology is used to document the data function rules, but the same approach can be applied with other techniques.

To identify data functions

a) identify all logically related and user recognizable data or control information within the counting scope,

b) exclude entities that are not maintained by any application,

c) group related entities that are entity dependent,

NOTE 2 Entities that are entity independent are separate logical groups of data.

d) exclude those entities below, referred to as code data:

1) substitution data entity that contains a code and an explanatory name or description;

2) single occurrence entity that contains one or more attributes that rarely change, if at all;

3) entity that contains data that is basically static or may change very rarely;

4) default values entity that contains values for populating attributes;

5) valid values entity that contains available values for selection or validation;

6) entity that contains a range of values for validation,

NOTE 3 These entities referred to above as code data may also contain other attributes for audit purposes and to define effective dates. The presence of these attributes does not alter the nature of these entities.

DRAFT 2008

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

e) exclude entities that do not contain attributes required by the user, and

f) remove associative entities that contain additional attributes not required by the user and associative entities that contain only foreign keys; group foreign key attributes with the primary entities.

NOTE 4 Foreign key attributes are data required by the user to establish a relationship with another data function.

3.4.3 Classify each data function as either an ILF or an EIF

Classify a data function as an

a) Internal Logical File (ILF) if it is maintained by the application being measured, or

b) External Interface File (EIF) if it is

- referenced, but not maintained, by the application being measured, and
- identified in an ILF in one or more other applications.

3 4 4. Count DETs for each data function

To count Data Element Types (DETs) for a data function

a) count one DET for each unique user recognizable, non-repeated attribute maintained in or retrieved from the data function through the execution of an elementary process,

b) count only those DETs being used by the application being measured when two or more applications maintain and/or reference the same data function,

NOTE Attributes that are not referenced by the application being measured are not counted.

c) count one DET for each attribute required by the user to establish a relationship with another data function, and

d) review related attributes to determine if they are grouped and counted as a single DET or whether they are counted as multiple DETs. Grouping will depend on how the elementary processes use the attributes within the application.

EXAMPLE The attributes (first name, middle initial, last name) are grouped and counted as

- name (first name, middle initial, last name) if these attributes are always used together,
- first names (first name and middle initial) and last name if, in addition, last name is sometimes used on its own, or
- first name, middle initial and last name, if all three can be used independently.

3 4.5 Count RETs for each data function

To count Record Element Types (RETs) for a data function

a) count one RET for each data function (i.e., by default, each data function has one sub-group of DETs to be counted as one RET),

b) count one additional RET for each of the following additional logical sub-groups of DETs (within the data function) that contains more than one DET:

1) associative entity with non-key attributes;

2) sub-type (other than the first sub-type);

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

3) attributive entity, in a relationship other than mandatory 1-1;

NOTE If you don't have a data model, look for repeating groups of data in order to identify RETs.

EXAMPLE 1 An invoice has a header section of the customer information and line items of the purchases (item number, description, price, weight, etc.). The Header section would be counted as one RET. The line items are a repeating group and would be counted as an additional RET.

EXAMPLE 2 A single repeating attribute such as multiple account numbers for a customer is not a repeating group and would be counted as a single DET and not as a RET.

EXAMPLE 3 A multiple occurring group of DETs such as year, month and budget amount is a repeating group but would be counted as three DETs and not as a RET.

3.4.6 Determine the functional complexity for each data function

Use the number of RETs and DETs to determine the functional complexity of each data function in accordance with Table 1.

Table 1 — Data function complexity

		DETs		
		1 – 19	20 – 50	> 50
RETs	1	Low	Low	Average
	2 – 5	Low	Average	High
	> 5	Average	High	High

3 4.7 Determine the functional size for each data function

Use the type and functional complexity to determine the functional size of each data function in accordance with Table 2.

Table 2 — Data function size

		Type	
		ILF	EIF
Functional complexity	Low	7	5
	Average	10	7
	High	15	10

3.5 Measure transactional functions

3 5.1 Overview

Transaction functionality satisfies the functional requirements that process data. All transactional functionality within the counting scope needs to be evaluated to identify each unique elementary process. Once an elementary process is classified, it is known as a transactional function.

DRAFT 2008

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

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

To measure transactional functions

a) identify each Elementary Process (EP) required by the user in accordance with 3.5.2,

b) classify each transactional function as an External Input (EI), External Output (EO) or an External Inquiry (EQ) in accordance with 3.5.3,

c) count the File Types Referenced (FTRs) for each transactional function in accordance with 3.5.4,

d) count the Data Element Types (DETs) for each transactional function in accordance with 3.5.5,

e) determine the functional complexity for each transactional function in accordance with 3.5.6, and

f) determine the functional size of each transactional function in accordance with 3.5.7.

3.5 2. Identify elementary processes

3.5.2.1 To identify each elementary process

a) Compose and/or decompose the functional requirements into the smallest unit of activity, which satisfies all of the following:

   1) is meaningful to the user;

   2) constitutes a complete transaction;

   3) is self-contained;

   4) leaves the business of the application being counted in a consistent state,

EXAMPLE 1 A functional requirement may state that a function must be provided to Maintain Employee information. That requirement is decomposed into smaller units of work such as Add Employee, Change Employee, Delete Employee, Inquire about Employee.

EXAMPLE 2 Individual requirements may state the need to add different types of employee information (e.g., address, salary and dependent information), but the smallest unit of activity meaningful to the user is to Add Employee.

b) Identify an elementary process for each unit of activity identified that meets all of the criteria in 3.5.2.1 a.

3.5.2.2 To determine unique elementary processes

a) When compared to an Elementary Process (EP) already identified, count two similar EPs as the same Elementary Process if they:

   1) require the same set of DETs, and;

   2) require the same set of FTRs, and;

   3) require the same set of processing logic to complete the elementary process (refer to the list below 3.5.2.2 b),

NOTE One elementary process may include minor variations in DETs or FTRs as well as multiple alternatives, variations or occurrences of processing logic below.

EXAMPLE 1 When an EP to Add Employee has been identified, it is not divided into two EPs to account for the fact that an employee may or may not have dependents. The EP is still Add Employee, and there is variation in the processing logic and DETs to account for dependents.

b) Processing logic to complete an elementary process are described below:

DRAFT 2008

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

Forms of processing logic

1) Validations are performed;

EXAMPLE 2 When adding a new employee to an organization, the employee process validates the employee type DET.

2) Mathematical formulas and calculations are performed;

EXAMPLE 3 When reporting on all employees within an organization the process includes calculating the total number of salaried employees, hourly employees and all employees.

3) Equivalent values are converted;

EXAMPLE 4 Employee age is converted to an age range group using a table.

4) Data is filtered and selected by using specified criteria to compare multiple sets of data;

EXAMPLE 5 To generate a list of employees by their assignment, an elementary process compares the job number of a job assignment to select and list the appropriate employees with that assignment.

5) Conditions are analyzed to determine which are applicable;

EXAMPLE 6 Processing logic exercised by the elementary process when an employee is added will depend on whether an employee is paid based on salary or hours worked. The entry of DETs (and the resulting processing logic) based upon a different choice (salary or hourly) in this example is part of one elementary process.

6) One or more ILFs are updated;

EXAMPLE 7 When adding an employee, the elementary process updates the employee ILF to maintain the employee data.

7) One or more ILFs or EIFs are referenced;

EXAMPLE 8 When adding an employee, the currency EIF is referenced for the correct US dollar conversion rate to determine an employee's hourly rate.

8) Data or control information is retrieved;

EXAMPLE 9 To view a list of employees, employee information is retrieved from a data function.

9) Derived data is created by transforming existing data to create additional data;

EXAMPLE 10 To determine (derive) a patient's registration number (e.g., SMIJO01), the following data is concatenated:

- the first three letters of the patient's last name (e.g., SMI for Smith);
- the first two letter of the patient's first name (e.g., JO for John);
- a unique two-digit sequence number (starting with 01).

10) Behavior of the application is altered;

EXAMPLE 11 The behavior of the elementary process of paying employees is altered when a change is made to pay them every other Friday versus on the 15th and the last day of the month; i.e., 26 vs. 24 pay periods.

11) Prepare and present information outside the boundary;

EXAMPLE 12 A list of employees is formatted and displayed for the user.

12) Capability exists to accept data or control information that enters the application boundary;

DRAFT 2008

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

EXAMPLE 13 A user enters information to add a customer order to the application.

13) Sorting or arranging a set of data. This form of processing logic does not impact the identification of the type or contribute to the uniqueness of an elementary process; i.e., the orientation of the data does not constitute uniqueness,

EXAMPLE 14 The list of employees is sorted in either alphabetical or location order.

EXAMPLE 15 On an order entry screen, the order header information is arranged at the top of the screen, and the order details are placed below.

c) Do not split an elementary process with multiple forms of processing logic into multiple elementary processes.

EXAMPLE 16 An elementary process that accepts and validates data from the user, reads and filters data from an ILF and sorts and presents the results back to the user cannot be split into multiple elementary processes.

3 5.3. Classify each elementary process as a transactional function

3.5.3.1 For each elementary process

a) identify the primary intent as one of the following:

1) altering the behavior of the application;

2) maintaining one or more ILFs;

3) presenting information to the user,

b) identify the forms of processing logic required to complete the elementary process from the list presented in 3.5.2.2 b).

3.5.3.2 Classify each elementary process as

a) an EI, if it:

1) includes processing logic to accept data or control information that enters the application boundary;

2) has a primary intent of either

i) maintaining one or more ILFs, or

ii) altering the behavior of the application,

b) an EO, if it has the primary intent of presenting information to the user and it includes at least one of the following forms of processing logic:

1) mathematical calculations are performed;

2) one or more ILFs are updated;

3) derived data is created;

4) the behavior of the application is altered,

c) an EQ, if it has the primary intent of presenting information to the user and it:

1) references a data function;

2) does not satisfy the criteria to be classified as an EO.

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

NOTE 1 Table 3 presents a summary of the relationship between the primary intent and transactional function type.

Table 3 — Relationship between primary intent and transactional function type

Function	Transactional function type		
	EI	EO	EQ
Alter the behavior of the application	PI	F	N/A
Maintain one or more ILFs	PI	F	N/A
Present information to a user	F	PI	PI

key

PI	the primary intent of the transactional function type
F	a function of the transactional function type that is sometimes present but which is not the primary intent
N/A	the transactional function type is not allowed to perform this function type

NOTE 2 Table 4 presents a summary of the relationship between forms of processing logic and transactional function type.

Table 4 — Relationship between processing logic and transactional function type

Form of processing logic:	Transactional function type		
	EI	EO	EQ
1. Validations are performed	c	c	c
2. Mathematical calculations are performed	c	m*	n
3. Equivalent values are converted	c	c	c
4. Data is filtered and selected by using specified criteria to compare multiple sets of data	c	c	c
5. Conditions are analyzed to determine which are applicable	c	c	c
6. At least one ILF is updated	m*	m*	n
7. At least one ILF or EIF is referenced	c	c	m
8. Data or control information is retrieved	c	c	m
9. Derived data is created	c	m*	n
10. Behavior of the application is altered	m*	m*	n
11. Information is prepared and then presented outside the boundary	c	m	m
12. Data or control information entering the application boundary is accepted	m	c	c
13. A set of data is sorted or arranged	c	c	c

key

m	it is mandatory that the transactional function type perform the form of processing logic
m*	it is mandatory that the transactional function type perform at least one of these (m*) forms of processing logic
c	the transactional function type can perform the form of processing logic, but it is not mandatory
n	the transactional function type cannot perform the form of processing logic

3 5.4 Count FTRs for each transactional function

For each transactional function, count one FTR for each unique data function that is accessed (read from and/or written to) by the transactional function.

3.5.5 Count DETs for each transactional function

To count DETs for a transactional function

a) Review everything that crosses (enters and/or exits) the boundary,

DRAFT 2008

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

b) Count one DET for each unique user recognizable, non-repeated attribute that crosses (enters and/or exits) the boundary during the processing of the transactional function,

EXAMPLE 1 DETs that cross the boundary include

- attributes the user enters via a screen as well as those displayed on a report or screen,
- attributes that enter the application boundary that are required to specify when, what and/or how the data is to be retrieved or generated by the elementary process,
- attributes provided by, or presented to, the user of the transactional function, and
- attributes in an electronic file that enter or exit the boundary.

c) Count only one DET per transactional function for the ability to send an application response message even if there are multiple messages,

EXAMPLE 2 If multiple error/confirmation messages are presented to the user, only one DET is counted.

d) Count only one DET per transactional function for the ability to initiate action(s) even if there are multiple means to do so, and

EXAMPLE 3 If the user can initiate the generation of a report by clicking on the OK button or by pressing a PF key, only one DET is counted.

e) Do not count the following items as DETs:

- literals such as report titles, screen or panel identifiers, column headings and attribute titles;
- application generated stamps such as date and time attributes;
- paging variables, page numbers and positioning information; e.g., 'Rows 37 to 54 of 211';
- navigation aids such as the ability to navigate within a list using "previous", "next", "first", "last" and their graphical equivalents;
- attributes generated within the boundary by a transactional function and saved to an ILF without exiting the boundary;
- attributes retrieved or referenced from an ILF or EIF for participation in the processing without exiting the boundary.

3.5.6 Determine the functional complexity for each transactional function

Use the number of FTRs and DETs to determine the functional complexity for each transactional function in accordance with Table 5 or Table 6.

DRAFT 2008

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

Table 5 — EI functional complexity

		DETs		
		1 – 4	5 – 15	> 15
FTRs	0 – 1	Low	Low	Average
	2	Low	Average	High
	> 2	Average	High	High

Table 6 — EO and EQ functional complexity

		DETs		
		1 – 5	6 – 19	> 19
FTRs	0 – 1	Low	Low	Average
	2 – 3	Low	Average	High
	> 3	Average	High	High
NOTE An EQ has a minimum of 1 FTR.				

3.5.7 Determine the functional size of each transactional function

Use the functional complexity to determine the functional size of each transactional function in accordance with Table 7.

Table 7 — Transactional function size

		Type		
		EI	EO	EQ
Functional complexity	Low	3	4	3
	Average	4	5	4
	High	6	7	6

3.6 Measure conversion functionality

The counting scope of a development or enhancement project may also include the functional size of conversion functionality required for the project. Count conversion transactional functions and data functions (that have not already been counted) in accordance with 3.4 and 3.5.

NOTE A requirement for conversion functionality does not mean that it has to be measured. It is the counting scope that dictates what functionality is counted.

3.7 Measure enhancement functionality

Enhancement projects can involve adds, changes to and deletion of existing functionality.

a) Do not modify the boundary already established for the application(s) being modified.

b) Count data functions that are added, changed or deleted in accordance with 3.4.

c) Count transactional functions that are added, changed or deleted in accordance with 3.5.

d) The application functional size may be updated to reflect:

1) added functionality, which increases the application functional size;

DRAFT 2

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

2) changed functionality, which may increase, decrease or have no effect on the application functional size;

3) deleted functionality, which decreases the application functional size.

NOTE 1 A change to a data function may involve adding, changing or deleting of DETs and/or RETs.

NOTE 2 A change to a transactional function may involve adding, changing or deleting of DETs and/or FTRs and/or changing of processing logic.

3.8 Calculate functional size

Consider the purpose and counting scope when selecting and using the appropriate formula to calculate the functional size.

To calculate a development project functional size, use Formula (1):

$$DFP = ADD + CFP \quad (1)$$

where

DFP is the development project function point count

ADD is the size of the functions to be delivered to the user by the development project

CFP is the size of the conversion functionality

To calculate an application functional size from a measurement after the development project or at any time during the application's life cycle, use Formula (2):

$$AFP = ADD \quad (2)$$

where

AFP is the application function point count

ADD is the size of the functions to be delivered to the user by the development project (excluding the size of any conversion functionality) or the functionality that exists whenever the application is counted

To calculate an enhancement project functional size, use Formula (3):

$$EFP = ADD + CHGA + CFP + DEL \quad (3)$$

where

EFP is the enhancement project function point count

ADD is the size of the functions being added by the enhancement project

CHGA is the size of the functions being changed by the enhancement project – as they are / will be after implementation

CFP is the size of the conversion functionality

DEL is the size of the functions being deleted by the enhancement project

To calculate an application functional size after an enhancement project, use Formula (4):

$$AFPA = (AFPB + ADD + CHGA) - (CHGB + DEL) \quad (4)$$

DRAFT 2006

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

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale
© ISO/IEC 2004 – All rights reserved

where

AFPA is the application function point count after the enhancement project

AFPB is the application function point count before the enhancement project

ADD is the size of the functions being added by the enhancement project

CHGA is the size of the functions being changed by the enhancement project – as they are / will be after implementation

CHGB is the size of the functions being changed by the enhancement project – as they are / were before the project commenced

DEL is the size of the functions being deleted by the enhancement project

3.9 Document the function point count

Document the following as part of the function point count:

— the purpose and type of count;

— the counting scope and application boundary;

— the date of the count;

— a list of all data and transactional functions, including their type and complexity and number of function points assigned;

— the result of the count (refer to 3.10);

— any assumptions made and issues resolved.

The documentation can also include the following:

— identification of the source documentation on which the count was based;

— identification of the participants, their roles and qualifications;

— for each data function, the number of DETs and RETs;

— for each transactional function, the number of DETs and FTRs;

— a cross-reference of all data functions to transactional functions;

— a cross-reference of all data functions to the related abstractions in the source documentation;

— a cross reference of all transactional functions to the related abstractions in the source documentation.

NOTE 1 Negotiate the level of documentation with the client, and inform the client about the related costs and benefits.

NOTE 2 A fully documented Function Point Count will facilitate traceability, usability and maintainability; however, a client may be interested only in the final result.

3.10 Report the result of the function point count

NOTE Consistently reporting the results of function point counts enables the readers to identify the standard to which they comply.

DRAFT 2008

--`,,```,,,,````-`-`,,`,,`,`,,`---
Copyright International Organization for Standardization ll rights reserved
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS Not for Resale

3.10.1 Report results that comply with this International Standard as follows:

S FP (IFPUG Y–V)

where

S is the result of the function point count

FP is the unit of size of the IFPUG FSM Method

Y is year of publication of this standard

V is version of the IFPUG Counting Practices Manual used, if any

EXAMPLE 1 250 FP (IFPUG-2008-4.3)

3.10.2 Report results that comply with a local customization of this International Standard as:

S FP (IFPUG Y–V–c)

where

c represents one or more characters indicating the result does not fully comply with this International Standard

EXAMPLE 2 250 FP (IFPUG-2008-4.3-a)

DRAFT 2008

Copyright International Organization for Standardization
Provided by IHS under license with ISO
No reproduction or networking permitted without license from IHS
© ISO/IEC 2004 – All rights reserved
Not for Resale

Annex A

(informative)

Consolidated complexity and functional size tables

For convenience, the complexity and functional size tables are repeated in Tables A.1 through A.5.

Table A.1 — Data function complexity

		DETs		
		1 – 19	20 – 50	> 50
RETs	1	Low	Low	Average
	2 – 5	Low	Average	High
	> 5	Average	High	High

Table A.2 — Data function size

		Type	
		ILF	EIF
Functional complexity	Low	7	5
	Average	10	7
	High	15	10

Table A.3 — EI functional complexity

		DETs		
		1 – 4	5 – 15	> 15
FTRs	0 – 1	Low	Low	Average
	2	Low	Average	High
	> 2	Average	High	High

Table A.4 — EO and EQ functional complexity

		DETs		
		1 – 5	6 – 19	> 19
FTRs	0 – 1	Low	Low	Average
	2 – 3	Low	Average	High
	> 3	Average	High	High
NOTE An EQ has a minimum of 1 FTR.				

Table A.5 — Transactional function size

		Type		
		EI	**EO**	**EQ**
Functional complexity	Low	3	4	3
	Average	4	5	4
	High	6	7	6

DRAFT 2008

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