FUNCTION POINT ANALYSIS

VALIDATING THE RESULT

P.M. Morris and J.M Desharnais
Thursday, July 19, 2001

Version 1.3

19 July, 2001

ABSTRACT

Function Point Analysis (FPA) is rapidly gaining worldwide acceptance as the most effective method for sizing software. Organisations are using the size of software measured in Function Points to assist them in the management and control of their software production environment. They use this measure of software size when estimating the cost, effort and duration of software projects. It is one of the parameters necessary for benchmarking productivity, performance and quality of software projects and their delivered products. However before data from Function Point Analysis counts can be used for comparison or prediction it must first be checked for accuracy and precision. This paper describes a method for checking the validity of the data collected during a Function Point count which has used the rules first described by Alan Albrecht in 1979 and later defined by the International Function Point Users Group (IFPUG). The paper identifies common sources of errors made during FPA and cites potential reasons for variances from the expected norm. The validation method has been developed by the authors based on their collective experiences of performing FPA and evaluating the results for over 300 software applications. They recommend that all results from Function Point Analysis be validated before being formally accepted and used by an organisation.

INTRODUCTION

The process currently used to validate Function Point counts relies on the experience of the individual rather than a prescribed scientific process. The objective of this paper is to document a more rigorous and quantitative approach based on our own experience in validating the results of FPA. It is hoped that by sharing our methods with others in the FPA community that they may contribute to the refinement of these validation techniques and that the improvement in the process of validating FPA counts will increase the reliability and acceptance of the FPA technique.

Validation of all Function Point counts is essential for an organisation to be able to confidently use them. It is necessary for counts to be accurate and consistent with respect to the interpretation of the counting rules for:

- a particular application,
- across applications,
- internally within an organisation and
- externally when comparing results with other organisations.

To help ensure the accuracy of a function Point count it needs to be checked both at the beginning of the counting process and at the end. At the beginning it needs to be checked when the purpose, boundary and scope have been identified and the strategy to be used for counting has been decided. Once the count has been completed the end result also needs to be validated using the process described within this paper.

Whilst we aim for an accurate and precise result, industry generally accepts that Function Point counts have a marginal relative error of plus or minus 10%.

These review procedures apply to counts performed using the rules and guidelines prescribed within the IFPUG Function Point Counting Practices Manual Release 4.0[i] and apply only to the *unadjusted* count. They exclude validation of the assessment ratings for the General Systems Characteristics.

1. FPA VALIDATION REVIEW

The recommended initial strategy for validating Function Point counts is to:

- prepare for the validation process by identifying the major participants in the count, and
- ensure that a complete set of information is available to the review .

Once the background of the count is established a high level review is performed aimed at identifying any major strategy errors employed in the count. These errors may be either in the overall approach or the participants understanding of the technique or the software application.

If the count passes this initial high level review then it is further investigated by comparing the summary of the count results with those predicted by counts for other applications.

If any inconsistencies are identified they are analysed for their origin.

If the count compares unfavourably with the predicted results or the variances cannot be sufficiently explained, then the next step is an in-depth examination of the detailed count focusing on areas most likely to contribute to errors.

If the count passes the high and intermediate level reviews, then a sample set of functions is investigated in detail.

Each level of the review process examines properties of the count and/or the attributes of the software being measured.

- The *high* level review examines the :
  - ◊ unadjusted functional size of the software being measured,
  - ◊ scope of the Function Point count,
  - ◊ boundary to the software being measured.
- The *intermediate* level review examines the :
  - ◊ relationship between the number of logical files and the unadjusted functional size of the software,
  - ◊ relationship between the numbers of functions to each other,
  - ◊ percentage contribution of each function type to the unadjusted functional size of the software,
  - ◊ profile of the complexity of the functions.
- The *lowest* level review examines the :
  - ◊ function point count of selected Data Business Functions,
  - ◊ function point count of selected Transaction Business Functions.

This validation process relies on techniques which highlight exceptions from the norm. It has proved to be successful in quickly detecting incorrect function point counts for software. However there could be instances where a valid count is highlighted as being incorrect. This will occur where the software being measured does not follow normal expected patterns of behaviour or the software has been incompletely specified. Alternatively be aware there could be situations where the FPA counting rules have been applied incorrectly and the count is invalid but the attributes of the count could still remain within the bounds of the exception conditions. If this occurred the count would not be highlighted as being incorrect at the high and intermediate levels of the review. However these errors should be detected at the detailed level of the review.

1.1. Preparation for Review

This step establishes the background of the count. The following information whilst not essential to the review process will assist in the assessment of the count :

- **Count Environment**
  - ◊ the names of the count participants and their roles in the FPA count.
    This enables the reviewer to schedule the key people to be available during the review to answer questions.
  - ◊ the names of the people who may provide expert knowledge about the application.
    The reviewer may need to reference additional sources of information in order to check the validity of particular counting decisions.
  - ◊ the source of information and documentation on which the count was based.
    This enables the reviewer to assess the validity of the count with respect to the content, quality (completeness and correctness) and version of information available at the time of counting rather than validating it with respect to the current version of that information.
  - ◊ the version of the FPA Counting Guidelines used.
    If the count has been performed using an obsolete version of FPA counting guidelines then it may be more appropriate to first update the count according to the correct standards before reviewing it.
  - ◊ the purpose for performing the FPA count.
    This enables the reviewer to check that the type of count, the application boundary and the scope of the count are appropriate for how the resulting count will be used.
  - ◊ a brief description of the type of functionality delivered by the software.
    This enables the reviewer to gain an understanding of what is being reviewed and assess whether the FPA Guidelines used were appropriate for this type of software and if some of the validation techniques used in the review are applicable.

- **Application Boundary**
  - ◊ a context diagram which illustrates how this software interacts with other software is also useful.
    This enables the reviewer to determine if the application boundary has been positioned correctly.

- **Detailed Count Results**

The following information is essential for the review:

- ◊ Count Description : identifying the type of count (application count, development project count or enhancement project count), the date it was performed and the stage in the lifecyle of the software when the count was performed.
- ◊ Count Summary : a summary of the Function Point count stating for each *type* of function. (eg. External Inputs, External Outputs, External Inquiries, Internal Logical Files, External Interface Files) the total number of functions within each level of complexity rating (eg. low , average or high).
- ◊ Transaction and File List : including the name, type and complexity rating of each Transaction and Data function.
- ◊ Count Notes : including any assumptions or comments which further clarify any counting decisions.
- ◊ Supporting Documentation : documentation which will enable the reviewer to assess if the Data functions and Transaction functions have been correctly assessed. eg. Requirements Specification, Logical Data Model, Functional Specification and/or User Manual.

- **Other Metrics for the software being sized**

The following metrics are optional but will assist with the review:

- ◊ Effort hours consumed by the development of the software ,
- ◊ Number of staff required to support the software,
- ◊ Number of pages of documentation used to specify the software's characteristics.

1.2. High Level FPA Review

1.2.1. Examine the Unadjusted Size of the Software

This step checks the reasonableness of the functional size by comparing it to that predicted by other attributes of the software. These attributes include:

- productivity rates collected for the software's development and support activities,
- the type of business addressed by the functionality delivered by the software.

NOTE: *Other researchers[ii] have identified a relationship between the lines of code (KLOC) used to write the software and its functional size measured in function points. In our experience this method of 'backfiring' lines of code to predict functional size is not suitable for validation of Function Point counts because it produces very inconsistent results.*

1.2.1.1. Validation Steps

- **Compare the functional size to that predicted by other metrics**

  Use metrics collected from other applications or projects from your organisation or published industry figures[iii] to predict the size of the software under review. For example:

  ◊ development effort hours per function point,

  eg. The International Software Benchmarking Standards Group Repository (Release 2 October 1995) reports an average delivery rate for Mainframe, 3GL applications of 8.9±5 hours per Function Point. If the application under review was built using a 3GL language and consumed 8900 effort hours to build then it would be expected to be around 1000 function points.

  ◊ delivered function points supported per person,

  eg. The following graph illustrates the results of a survey [iv] of 3 organisations and 8 production applications, most of which were less than 3 years old and all were developed for a mainframe COBOL, DB2 environment. Each person in the maintenance area supported on average about 1050 unadjusted function points. If the application under review has similar attributes and was supported by 2 people then it would be expected to be about 2,000 function points.

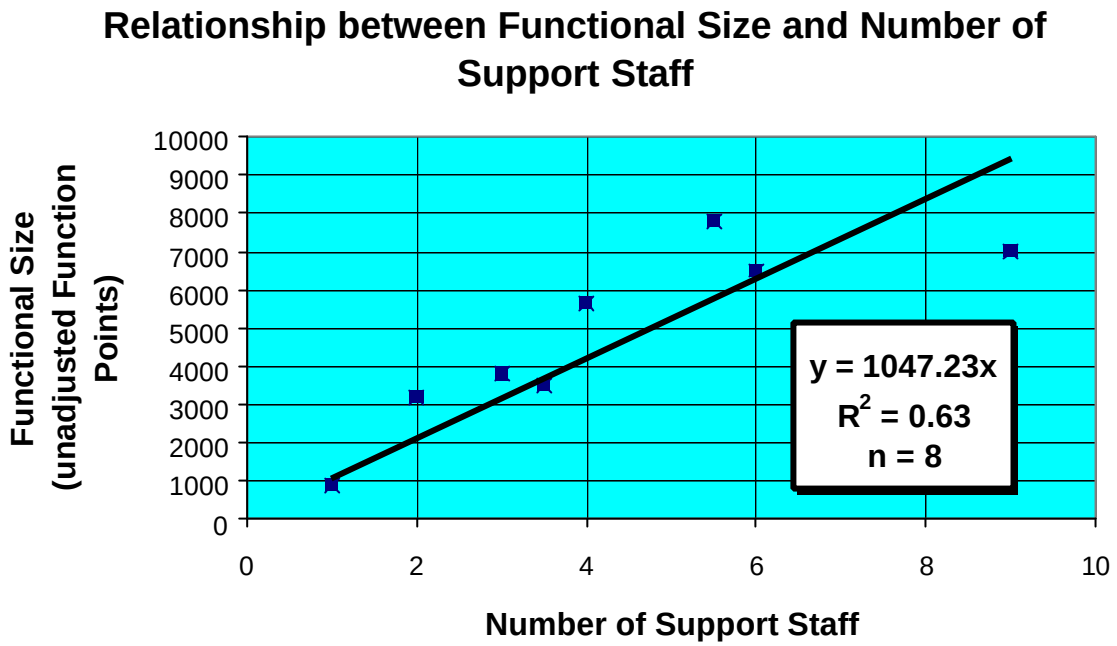

Figure 1

◊ function points specified per page of documentation.

eg. Requirements Specifications for three different ordering applications for three organisations were all written to IEEE Standard 830-1993. The number of function points specified for each application was 105, 118 and 295 and the number of pages of specifications were 16, 22 and 48 respectively. The trend for these applications is that they specify 6.1±0.6 function points per page of Requirements Specification. If the Requirements Specifications for the ordering application under review is 40 pages then its size would be predicted to be about 250 function points.

- **Compare Size predicted by other similar Applications**

  Common business areas (eg. payroll, personnel, accounts, sales, purchasing) within similarly sized organisations tend to operate in a similar manner and require comparable amounts of functionality to support their business processes. Use the known size of other similar applications to predict the size of the software under review.

  eg. The measured size of the Accounts Payable applications for four large manufacturing corporations was 1430, 1389, 1683 and 1528 unadjusted function points. If the primary function of the application under review was to also deliver Accounts Payable functionality to a large manufacturing corporation then its functional size would be expected to be between about

1200 to 1800 unadjusted function points.

1.2.1.2. Warning Signs:

If the size predicted by one or more of the above attributes of the software is outside the expected range by ±30% then the count may be incorrect.

1.2.2. Examine the Boundary of the Software

1.2.2.1. Overview

FPA sizes software from the logical external user view[v]. It only includes functions which are user identifiable and excludes those introduced by implementation methods or the technical environment. It measures the functionality delivered by user transactions which move data across the *boundary* between the software and the external user domain. Where the *boundary* is a conceptual interface between the software under study and its users[vi]. Where the 'User' may be any person, thing or other software application which uses or supplies the data. The *boundary* therefore acts as the 'membrane' through which the Transaction functions must permeate.

FPA does not measure data movements internal to the software which do not cross the *boundary* so the correct positioning of the *boundary* is critical to the validity of the result.

If the boundary is positioned incorrectly ie. along technical or physical limits then a logical group of business functions may be split across two or more software 'applications'. This results in a single business transaction being broken into several data movements between these 'applications' and consequently being incorrectly counted as several elementary processes.

1.2.2.2. Validation Steps

- Examine the context diagram. Determine if the boundary has been positioned along the natural confines of business based requirements rather than along the limits of the technical environment.
  - ⇒ The boundary should be positioned between:
    - ◊ distinct business applications *eg. Personnel System and the Sales System*,
    - ◊ the human user of the software and the software manifestation. *eg. screen, printed paper*.
  - ⇒ The boundary may be incorrect if placed between:
    - ◊ hardware platforms *eg. client and server platforms, local terminals and remote processors*,
    - ◊ software platforms *eg. between online and batch processing functions or between software coded in different languages*,
    - ◊ sub-sets of a business application *eg. between the module which reports on information gathered by the other functions in the application and those functions which maintain and store the information.*

- Examine the detailed count results.
  - ⇒ The boundary may be incorrect if:
    - ◊ external files account for more than 50% of the function points contributed by data functions *eg. this occurs when a sub-system is counted as a separate application but is very closely coupled with other sub-systems in the same business area.*
    - ◊ transactions input data which is then transmitted through the application boundary to another application without updating permanent Internal Logical Files. *eg. PC front-end to mainframe software.*

1.2.2.3. Action

Relocate the boundary ignoring the technical and physical limits. Locate it from the business requirements perspective, so that business transactions can execute to completion and achieve their business goal within the same software application.

Once the application boundary has been correctly placed, delete any duplicated files and internal data movements from the count.

1.2.3. Examine the Scope of the FPA

1.2.3.1. Overview

FPA can be used to size sub-sets of the software under study. These sub-sets of functions define the *scope* of the Function Point count. The grouping of functions within these sub-sets is determined by the purpose for the measuring the size of the software.

1.2.3.2. Validation Steps

Assess whether the functions included within the scope of the FPA count contribute to the needs identified in the purpose for performing the FPA.

eg. If the purpose for determining the size of the application is to use the result to determine replacement cost, then the scope of the count should only include those functions being replaced. Duplicated and obsolete functions are outside the scope and should be excluded.

eg. If the purpose for determining the size of the enhancement project is to use the result for estimating, then the scope of the count should only include those functions which are created, modified or deleted by the activities of the enhancement project. Common counting errors are :

- *to include within the scope of the enhancement count existing data functions which are accessed by the new functions but themselves not changed,*
- *to exclude functions which are changed by the project activities but their function point count remains the same.*

1.2.3.3. Action

Use the purpose and the type of FPA count to identify which functions should be included or excluded from the scope and adjust the count accordingly.

1.3. Intermediate Level Review

1.3.1. Overview

All 'systems' (eg. ecological, biochemical or business) accept input, process it and produce output. The amount of 'information' stored and processed is a function of what is input and what is required to be output. This relationship holds true for software information systems where the amount of data input is related to the amount of information stored and subsequently extracted.

This level of the review explores these intrinsic relationships between inputs, files and outputs/Inquiries for the measured software and compares them with values collected from industry. These review processes are only applicable when validating function point counts of software which satisfies the following criteria:

- is *data rich* rather than *process rich.* ie. typical Management Information System (MIS) type software.
- can operate as a 'self contained system' with minimal interfaces . That is the system has a balance of inputs, processing (access to stored data) and outputs.
- the 'application boundary' has been correctly placed according to the external business view rather then using technical , implementation or support boundaries.
- is an application or new development project. Only 'complete' stand alone functional groups can act as a 'system'.

NOTE: Enhancement Project counts which do not impact a discrete module of an application are excluded from this level of the review because the types of functions they impact are very variable. They cannot be compared with development or application counts because their functions do not have the expected proportions of inputs, processing and outputs.

1.3.2. Examine the Relationship between the Number of Logical Files and the Unadjusted Functional Size.

1.3.2.1. Overview

Since the number of functions to input and extract data is related to the amount of stored data then it follows that the total size in unadjusted function points should be a function of the amount of data used by the system. The following graphs confirm this hypothesis. They plot the total functional size in unadjusted Function Points to the:

- total number of Internal Logical Files (Figure 1),
- total number of Internal Logical Files plus the total number of External Interface Files (Figure 2).

for 161 new development projects and implemented applications[vii].

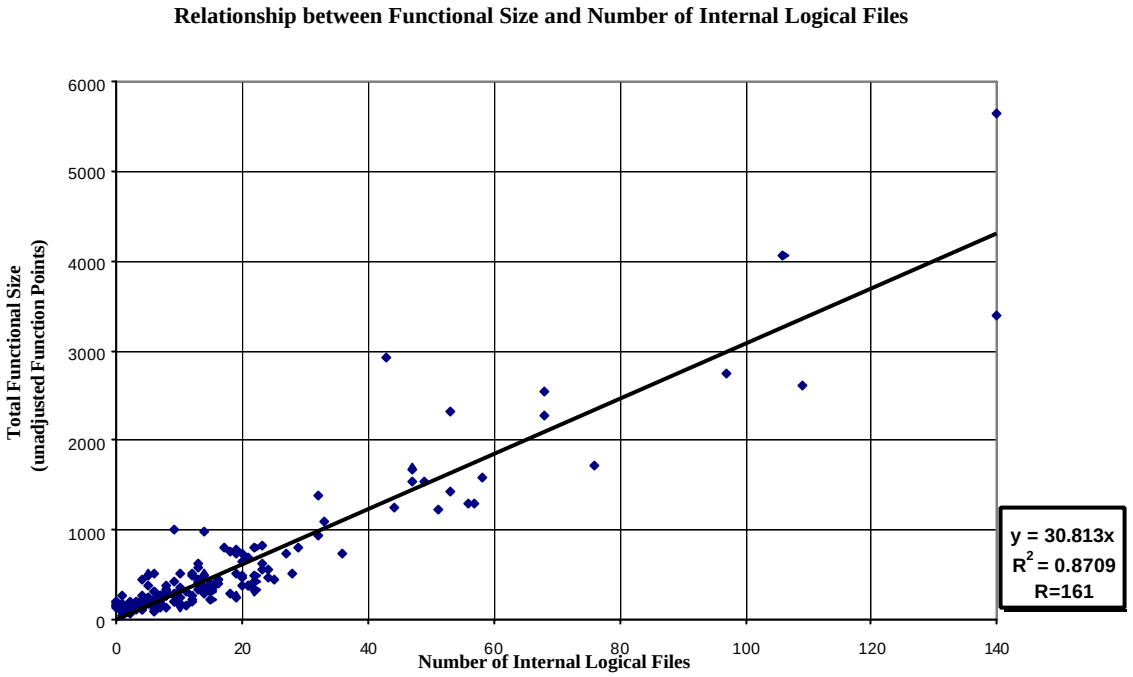

Figure 2

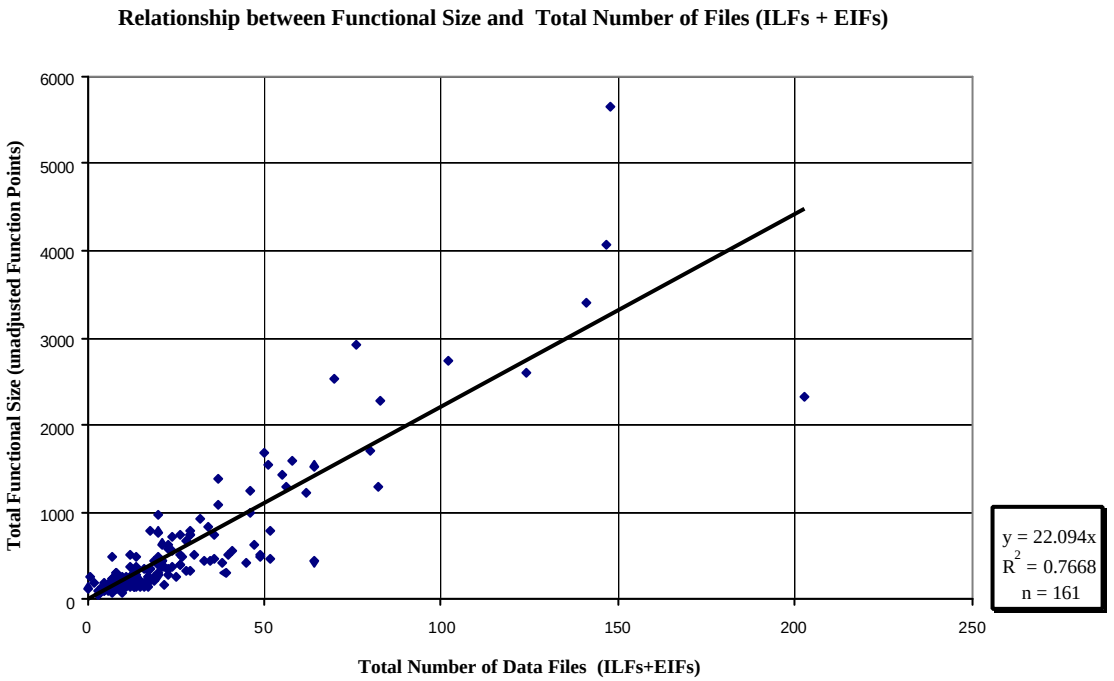

Figure 3

1.3.2.2. Validation Steps

Identify the number of Internal Logical Files (ILFs) and External Interface Files (EIFs) from the Count Summary information. Use the above graphs or your own organisation's figures to predict the size of the application under review.

eg. If the software being counted has :
⇒ *100 Internal Logical Files then its functional size is predicted to be about 3,000 unadjusted function points.*
⇒ *100 Internal Logical Files plus External Interface Files then its functional size would be expected to be about 2200 unadjusted function points.*

1.3.2.3. Warning Signs

If the functional size of the software is more than ± 20% different from that predicted by the graphs then it indicates that the count may be incorrect.

1.3.2.4. Action:

When performing the detailed review pay particular attention to checking that the Transactions and Logical Files have been correctly identified and correctly classified into type and complexity.

1.3.3. Examine the Relationship between the Numbers of Functions.

1.3.3.1. Overview

This check also explores the intrinsic relationship between data input, data stored and data output. The following graph plots the ratios of input and extraction (Output + Inquiry) transactional function types to the data files for above set of 161 applications and projects. It confirms the hypothesis that there is a balance between the *number* of :

- input functions used to maintain data,
- files being accessed, and
- output functions which extract the data.

The data illustrates a reasonably high correlation (r^2=0.8083) between the number of Inputs and the number of Internal Files. For each Internal File there is typically:

- 2.7 Inputs which maintain the data
- 1.2 Inquiries and
- 1.2 Outputs.

The correlation of Internal Files with extraction functions is lower. This may be because:

- reports for newly developed applications are usually limited to a minimum and older applications tend to have high numbers of reporting functions. The data represents a mixture of both types of counts , newly developed and older applications.
- it is sometimes difficult to distinguish between Outputs and Inquiries

The graphs below illustrate that the correlation between all types of transactions and files is higher for Internal Logical Files than for the combined total of ILFs and EIFs. For each File (ILF or EIF) there is typically:

- 1.9 Inputs
- 0.8 Inquiries and
- 0.9 Outputs.

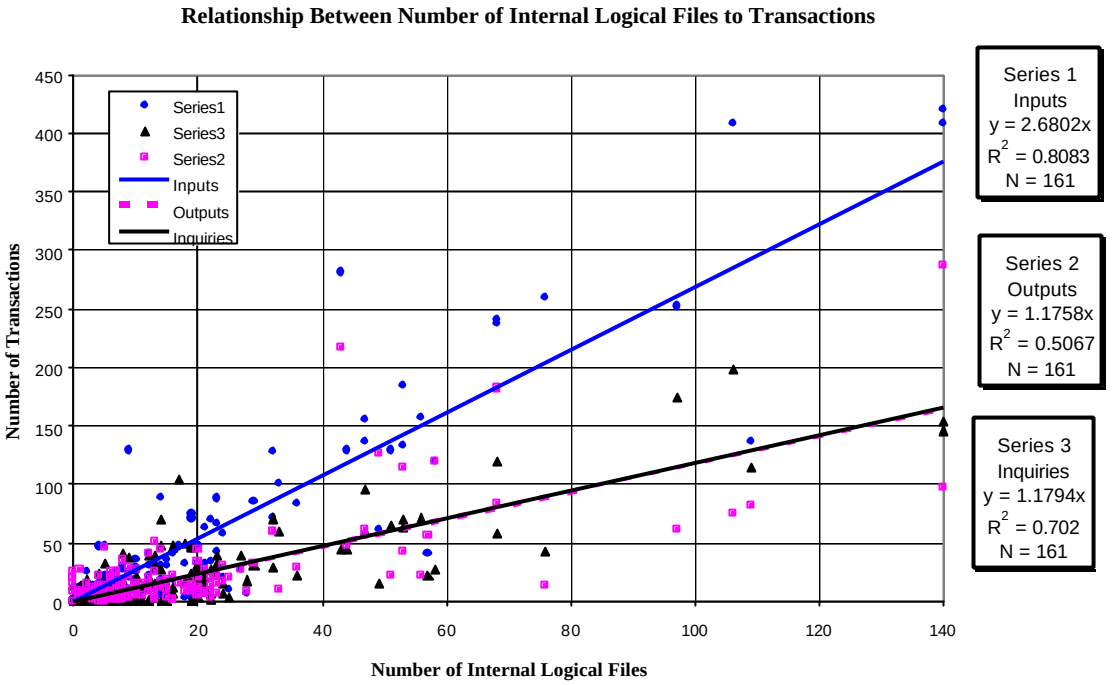

Figure 4

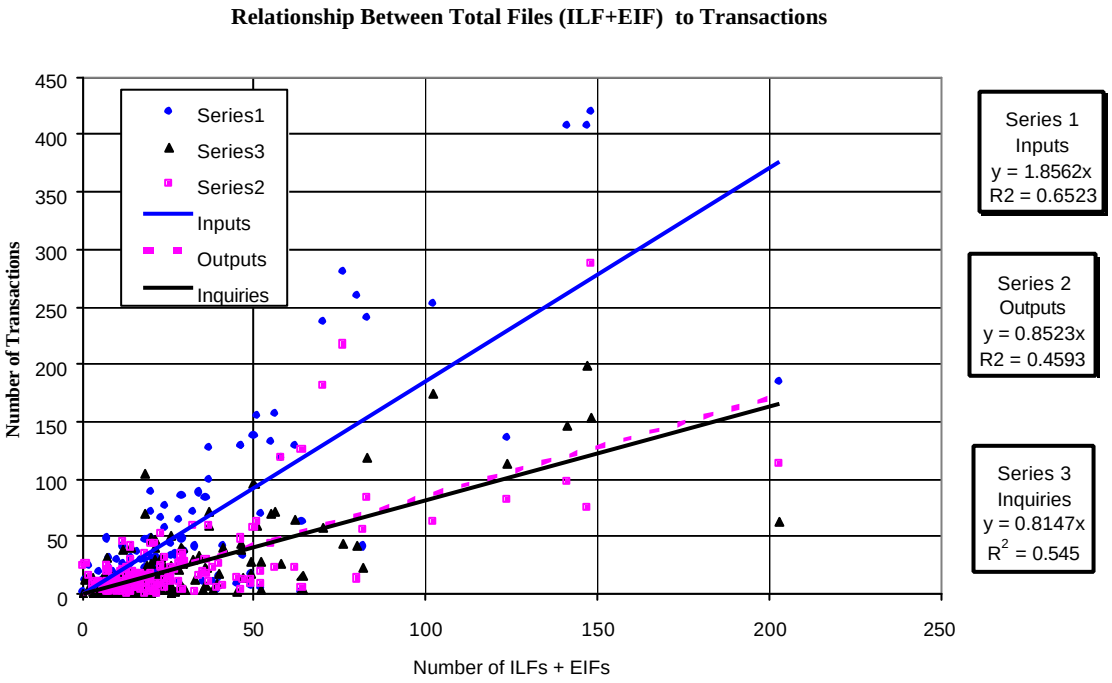

Figure 5

1.3.3.2. Validation Steps

Using the Count Summary information identify the number of functions for each function type. Check for potential errors in the count by comparing the ratio of the measured application's inputs and extraction functions to the number of files.

eg. If the application count being reviewed has 100 Internal Logical files then it can be expected to have about 270 input functions and 120 Output and 120 Inquiry functions.

eg. If the application count being reviewed has 100 files (internal logical files and External Interface files) then it can be expected to have about 180 input functions and 90 Output and 80 Inquiry functions.

1.3.3.3. Warning Signs

If the number of EIs , EOs or EQs is outside the normal range by more than $\pm$ 30% then check that the Transactions and Logical Files have been correctly identified and classified into type and complexity.

1.3.4. Examine the Percentage Contribution of Each Function Type to the Count

1.3.4.1. Overview

This check is similar to the previous one except it compares the percentage of function points contributed by each function type to the overall count. The percentage contribution of each function type has been found to be reasonably consistent for applications which are typical Management Information Systems.

The following doughnut chart illustrates the typical percentage contributions of functions. The outer ring is from the authors' database of counts and the inner ring from the International Software Benchmarking Database Standards Group Repository of 105 new development projects.[viii]. The percentage contribution of Internal Logical Files, Outputs and Inquiries is almost identical for both databases. The largest variations are for External Inputs and External Interface files. However, both databases indicate that about 24% of Function Points are contributed by Internal Files, between 12-14% by Inquiries, 22%-24% by Outputs, 26-39% by Inputs and 4%-12% by External Interface Files.

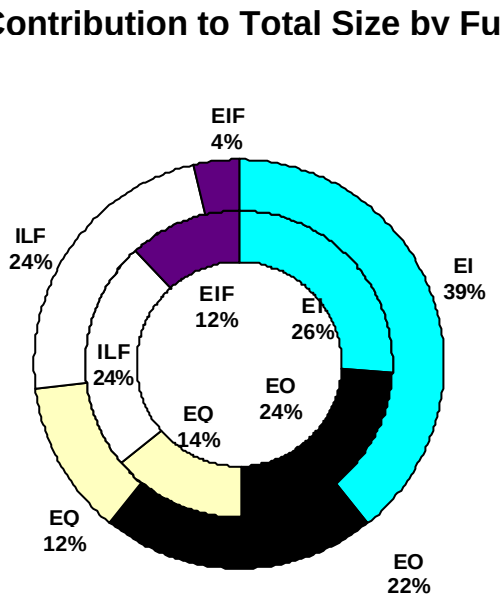

Figure 6 - Outer Ring = ISBG Database, Inner Ring = JMD , PM Database

1.3.4.2. Validation Steps

Use the Count Summary to calculate the percentage contribution of each function type to the total functional size and compare them with the above percentages.

1.3.4.3. Warning Signs

The count may be incorrect if the percentage contribution of function types is more than 5-10% outside the expected ranges. Check that the Transactions and Logical Files have been correctly identified and correctly classified into type and complexity.

1.3.5. Examine the Complexity of the functions

1.3.5.1. Overview

This check also explores the intrinsic nature of MIS applications. The following table compares the mean function points awarded for each function type found within the authors' database and the ISBGs database with the equivalent complexity rating in the IFPUG complexity matrices. In both databases the overall mean complexity for :

- ◊ Transactional function types is closest to 'average',
- ◊ Data function types is closest to 'low'.

Table 1

Function Type	Mean Function Points Awarded (Total number FPs /Total number functions)		Corresponding IFPUG Complexity Rating	
	ISBGs Data (n=106)	JDM, PM Data (n = 161)	Average	Low
Inputs	4.3	4.2	**4**	3
Outputs	5.4.	5.8	**5**	4
Inquiries	3.8	4.0	**4**	3
Internal Logical Files	7.4	7.8	10	**7**
External Interface Files	5.5	5.2	7	**5**

1.3.5.2. Warning Signs

Indications that the count may be incorrect are if the complexity of more than:

- 90% of EIs , 80% of EOs or 70% of EQs is outside the normal range,
- 10 % of Internal Logical Files or External Interface Files are 'average or high complexity'.

If the complexity of ***transactions* (*Elementary Processes)*** is outside the normal expected ranges then check that :

- the data elements counted only include those which should contribute to the complexity rating for that type of function,
- the logical files have been correctly grouped and are not causing each transaction to incorrectly reference too many or too few files.

If the complexity of ***logical files* (*ILFs or ElFs)*** is outside the normal expected ranges then check that :

- the data elements counted only include those which are accessed by transactions within this application,
- the data elements do not include foreign keys introduced purely for technical or navigation purposes,
- the sub-groupings of the data (record element types) are from a user perspective.

1.3.5.3. Action

If the files have been incorrectly grouped or an incorrect method for determining complexity has been used, then the software will need to be completely re-counted.

1.4. Lowest Level FPA Review

1.4.1. Overview

This level of the review is performed when the previous level reviews have assessed the count as being potentially :

- **valid** - then the detailed review is used to confirm the assessment, or
- **invalid** - then the detailed review is used to investigate sources of error to be corrected.

Randomly select a representative sample of business functions to be re-sized. Ensure that it includes all function types and has sufficient supporting documentation to enable assessment using FPA.

1.4.2. Examine the Data Business Functions

The correct identification of Logical data groups (Internal Logical files and External Interface files) is the most critical contributor in determining an accurate functional size for the software. If the data is grouped incorrectly to give too many or too few files then the complexity rating of every transaction accessing these files will also be incorrect.

1.4.2.1. Validation Steps

Check that the logical files (data functions) were interpreted from the Logical Data Model using the user business perspective.

If the number of function points contributed by ***External Interfaces*** is outside the normal expected ranges and is :

- Too **high,** then check for the following common counting errors or reasons for variances:
  - ◊ the boundary may have been placed around a group of functions which cannot act as a functional unit resulting in the need to access considerable amounts of external data,
  - ◊ transactional data files moving within the physical interface between applications have been incorrectly counted as External Interface Files.
- Too **low,** then check for the following common counting errors or reasons for variances:
  - ◊ external data access may have been incorrectly counted as an External Input or External Inquiry.
  - ◊ the application is completely stand alone (eg. PC Football Betting System) and does not have a requirement to interface externally.

If the number of function points contributed by ***Internal Logical Files*** is outside the normal expected ranges and is :

- Too **high,** then check for the following common counting errors or reasons for variances:
  - ◊ the information technology view of logical data has been used to identify files instead of the users business view. eg. all tables identified on the fully normalised data model have been identified as Internal Logical Files,
  - ◊ all sub-types of data have been counted as discrete Internal Logical Files,
  - ◊ physical data files have been counted as Internal Logical Files eg. Report Files created for performance reasons.
- Too **low,** then check for the following common counting errors or reasons for variances:
  - ◊ only highest level entities have been identified as Internal Logical Files. Sub-groupings of the file separately maintained by transactions have been incorrectly counted as Record Element Types rather than Internal Logical Files..
  - ◊ the logical files have been denormalised for performance or technical reasons and these physical files have been used as the basis for identifying Internal Logical Files. eg. single codes reference table file actually represents multiple logical files , one for each discrete type of business data separately maintained on the file.

1.4.2.2. Warning Signs

The count has a high probability of being incorrect if the detailed review of the names identifying the data functions :

- ◊ correspond directly to the physical implementation of the data storage. *eg. Customer database*;
- ◊ correspond directly to the fully normalised data model, *eg. they include intersecting entities with no non-key attributes,*
- ◊ are not on the logical data model:
  - *eg. files storing duplicated data (report files, temporary files , sort files),*
  - *eg. files which have been introduced for technical reasons (temporary files, backup files, archive files).*

1.4.2.3. Action

Re-count the files grouping the data from a logical user business perspective, then reassess the complexity of every transaction using the regrouped files.

1.4.3. Examine the Transaction Business Functions

The correct identification of elementary business processes is also critical to the correctness of the functional size measurement. The function point count will be more inaccurate if a function fails to be identified than if a function is identified and incorrectly classified for type and complexity.

1.4.3.1. Validation Steps

Check *how* the transactions were identified. ie. ensure that:

- each variation of a process was checked for differences in logical processing before being considered unique or just part of the elementary process.
- a logical business model of the application was used to identify transactions rather than basing their identification on a physical view. *eg. was the users business requirements used to identify functions or were they identified based solely on physical screens.*

If the number of function points contributed by ***External Inputs*** is outside the normal expected ranges and is:

- Too **high,** then check for the following common counting errors or reasons for variances:
  - ◊ input selection filters for reporting data may have been incorrectly counted as External Inputs,
  - ◊ physical data entry screens may have been incorrectly counted as logical transactions (elementary processes),
  - ◊ each slight variation in the data entered for an input transaction may have been incorrectly counted as a unique elementary process,

- ◊ physical transactions eg. save, exit, send may have been incorrectly counted as logical External Input transactions.

- Too **low,** then check for the following common counting errors or reasons for variances:
  - ◊ add, change and delete functions may have been bundled into a single maintenance input and not counted as discrete elementary processes,
  - ◊ counter may have not looked for additional input transactions beyond standard maintenance eg. C*ancel Invoice , Post Invoice to GL* etc,
  - ◊ an incoming file has been incorrectly counted as a single External Input instead of evaluating each transaction type for qualification as a unique elementary process,
  - ◊ physical files have been incorrectly grouped into logical files causing both the complexity of the files to be incorrect and the complexity of every transaction accessing the logical file to be incorrect,
  - ◊ business rules may not allow the user to delete data.

If the number of function points contributed by ***External Outputs*** is outside the normal expected ranges and is :

- Too **high,** then check for the following common counting errors or reasons for variances:
  - ◊ error and confirmation messages may have been incorrectly counted as outputs,
  - ◊ all elementary processes which calculate data have been incorrectly identified as External Outputs eg. *Add a New Order* calculates and displays the products discount price but it is an External Input not an output,
  - ◊ different formats of the same output not being investigated for being unique transactions *eg. printed version and the online print preview being counted as two discrete outputs*,
  - ◊ may be an older application which often have more reporting functions compared to new applications.
- Too **low,** then check for the following common counting errors or reasons for variances:
  - ◊ variations of a report with different logical processing may not have been counted as unique outputs. eg. Detail and Summary Reports,
  - ◊ outputs are typically low for projects counted early in the lifecycle since all reporting functions are often not identified by the user until later in the project.

If the number of function points contributed by ***External Inquiries*** is outside the normal expected ranges and is :

- Too **high,** then check for the following common counting errors or reasons for variances:
  - ◊ help Inquiries may have been incorrectly counted as different Inquiries for every External Input,

◊ implicit queries on stored data during maintenance functions may have been incorrectly counted as Inquiries. ie. incorrectly counted an implicit Inquiry on the data just prior to deleting it.

- Too **low,** then check for the following common counting errors or reasons for variances:
  - ◊ application may have a focus on monitoring or reporting and data extraction which requires more than simple retrieval of stored data,
  - ◊ mostly batch application which tends to retrieve and report on manipulated information rather than ad hoc queries on data as stored.

1.4.3.2. Warning Signs

The count has a high probability of being incorrect if the detailed review of the names identifying the transaction functions:

- directly correspond to screen names *eg. Order Header Screen; Logon Screen,*
- imply internal processes *eg. Validate Order, Process Price Calculation,*
- imply technical implementation functions *eg. Overnight Batch Run, Menu Selection Inquiry, Backup database.*

1.4.3.3. Action

If the transactions within the sample set of functions have been incorrectly assessed then the software needs to be re-counted

1.5. Complete the Review

If the Function Point count **passed** all levels of the review and is assessed as being **valid** then:

- mark the count as 'passed', with your name and date of the review,
- document any variances from the predicted values. (These are used as reference information to identify reasons for variances in future reviews.)

If the Function Point count **failed** the review and was assessed as being **invalid** then :

- document where errors (if any) were identified so they may be remedied, and avoided in the future,
- document reasons why errors were made. (This will assist FPA trainers to focus on these areas when teaching FPA.)
- schedule to review the count again after it has been corrected.

2. EXAMPLE REVIEW RESULTS

The following examples have been collected from across the industry and illustrate some of the potential counting errors and validation issues discussed within this paper for the first two levels of the review. It is recommended that each organisation also construct its own set of comparative data and review examples.

2.1. Example 1.

2.1.1. Count Summary

Example 1	LOW	AVERAGE	HIGH	TOTAL	%
EI	3	12	60	75	30%
EO	4	25	28	57	23%
EQ	3	15	30	48	19%
ILF	56	0	0	56	22%
EIF	10	5	0	15	6%
TOTAL	**76**	**57**	**118**	**251**	**100%**

2.1.2. Review Discussion

This count is from a small development project for a module which is part of a larger application. The type of application was not recorded with the count details.

- ElFs represent approximately 1/4 of the ILFs which is within the acceptable range.
- the combined contribution of the ElFs and the ILFs represent 28% of the total of function points which is within the acceptable range of 20% - 40%.
- EIs represent 30% of the total which is also within the accepted range of 30% - 60% .
- the combined contribution of the EOs and EQs is within the 35% - 40% accepted range.

2.2. Example 2.

2.2.1. Count Summary

Example 2	LOW	AVERAGE	HIGH	TOTAL	%
EI	180	92	171	443	19%
EO	400	45	21	466	20%
EQ	112	48	90	250	11%
ILF	1050	20	0	1070	46%
EIF	100	0	0	100	4%
TOTAL	**1842**	**205**	**282**	**2329**	**100%**

2.2.2. Review Discussion

This count is from an application which has as its main role 'data capture'. It illustrates the implications of the positioning of the boundary.

- ElFs represent less than 10% of the ILFs which is normal for a data capture system.
- EIs represent about 40% of the total function points for transactions which is in the range of 30% - 60% for EIs.
- Transactions represent only 50% of the function points which is outside the expected range of 60% - 80% found in an average application. This variation can probably be explained by the type of application which was counted. The boundary had been placed between the data capture component and other functions of a much larger application. If the whole application had been included in the count then the balance between transactions and files would more closely resemble the expected ratios.
- ILFs contribute more than the expected percentage of function points ie. 46% compared to the expected contribution of 24%. ILFs have most probably been over-counted due to the artificial separation of the data capture functions from the rest of the application.

The position of the application boundary should be further investigated.

2.3. Example 3

2.3.1. Count Summary

Example 3	LOW	AVERAGE	HIGH	TOTAL	%
EI	0	0	48	48	13%
EO	27	150	150	327	87%
EQ	0	0	0	0	0%
ILF	0	0	0	0	0%
EIF	0	0	0	0	0%
TOTAL	**27**	**150**	**198**	**375**	**100%**

2.3.2. Review Discussion

Examination of the results of this count immediately highlight potential counting errors.

- the count has not included any ILFs and EIFs. The boundary and scope are most likely incorrect since software must include at least one ILF to be considered an application.
- 8 high EIs (48 points) are included in the count. This highlights a potential counting error since the count does not include any ILFs for the EIs to update.

The severity of the counting errors suggest that the position of the boundary should be revised and the software recounted.

2.4. Example 4

2.4.1. Count Summary

Example 4	LOW	AVERAGE	HIGH	TOTAL	%
EI	1200	700	2500	4400	39%
EO	3000	480	900	4380	39%
EQ	0	0	0	0	0%
ILF	1400	10	0	1410	12%
EIF	1000	35	80	1115	10%
TOTAL	**6600**	**1225**	**3480**	**11305**	**100%**

2.4.2. Review Discussion

This count is for a very large application which processed a large amount of control information in batch mode compared with the number of interactive human user transactions. The boundary was positioned so that the computer acted as a 'user' and triggered transactions to process.

- EIs are at the high end of the expected percentage range 26-39%. The total number of EIs is 517and ILFs 200. This gives a ratio of 2.58 inputs to files. This is close to the predicted ratio of 2.68. EIs are probably correctly counted.
- The total contribution of function points contributed by files is 22% which is in the accepted range of 10-30%.
- EOs percentage contribution at 39% is higher than the expected percentage of 22-24%. However since there are no Inquiries (as can be expected in a primarily batch system) a higher percentage of outputs could be expected.
- The total number of files (407) predicts the total size to be about 9,000 function points. In our experience most MIS applications are smaller than 5000 function points. The size of 11,000 function points is higher than expected but within the 30% allowed error margin. This application size is not unusual for a large batch application.

Check that the overall size is correct. Investigate that the files and transactions have been correctly identified and assessed.

2.5. Example 5

2.5.1. Count Summary

Example 5	LOW	AVERAGE	HIGH	TOTAL	%
EI	30	40	60	130	10%
EO	160	125	280	565	44%
EQ	150	240	78	468	36%
ILF	77	0	0	77	6%
EIF	50	0	0	50	4%
TOTAL	**467**	**405**	**418**	**1290**	**100%**

2.5.2. Review Discussion

This count if for is a typical Human Resources Application for a medium sized organisation. It interfaces with the Payroll application and accesses many of its files. The results indicate that:

- there is large proportion of EOs and EQs. The high number of EOs and EQs is due to the application having multiple reports and multiple ways to inquire on the ILFs. Often this type of application includes a report generator. In these cases the number of EO s and EQs is often much lower than found in this application.
- there are relatively few ILFs and EIFs. The interface with the Payroll application explains the high proportion of EIFs to ILFs. The function points contributed by EIs (10%) and the total number of EIs compared to ILFs is less than expected.

[i] International Function Point User Group - Function Point Counting Practices Manual, Release 4.0 January 1994.
[ii] Jones Capers, " Programming Productivity" , McGraw-Hill, New York, NY, 1986.
[iii] International Software Benchmarking Standards Group (ISBG) Repository
[iv] Morris, P.M "Getting Projects Back on Track" IFPUG Spring 1994 Conference Proceedings
[v] St-Pierre, Denis, Desharnais Jean-Marc, Abran Alain, Gardner, Barbara, Definition of when requirements should be used to count Function Points in a project, Draft document, 1995
[vi] ISO/IEC/JTC1/SC7/DIS 14143 Definition of Functional Size Measurement.
[vii] Data Collected by Jean Marc Desharnais and Pam Morris. Results from 161 development project and application counts sourced from 14 Canadian and 6 Australian organisations
[viii] International Software Benchmarking Standards Group (ISBG) Repository.