Counting Infrastructure Software

Dr. Anthony L Rollo, SMS Ltd, Christine Green EDS

Many function point counters and managers of software counts believe that only whole applications may be sized using the IFPUG method. This is sometimes cited by some other methods as a reason for not using IFPUG. I and some colleagues have however been required by our clients to arrive at sizes for major software components such as the HCI component, the Server component and other software items often regarded as infrastructure.

This talk will outline the method of undertaking such counts. It will also deal with the important issues of identifying users, boundaries and so on from counting such software. The talk will also deal with the reasons why such counts are useful to the software developers. There can be pitfalls when reporting such counts to clients and counters need to be aware of these so they may avoid them.

1. Introduction

Infrastructure software is defined as: - Software that provides a variety of services to a range of software applications. In general infrastructure software does not communicate directly with the user, though it provides services ultimately used by the user. Within this category I include the major software components (Fig1.), which make up an application, such components may be HCI, or server level software. I also include software that provides the service level components, usually service level components serve the needs of several applications (Fig 2.) infrastructure components can act as communications interchange between components, as workflow elements in a workflow managed system such as customer relationship management. Infrastructure forms a major part in the Service Oriented Architecture which is becoming an increasingly important area of software development. Some forms of infrastructure software are known as middleware.

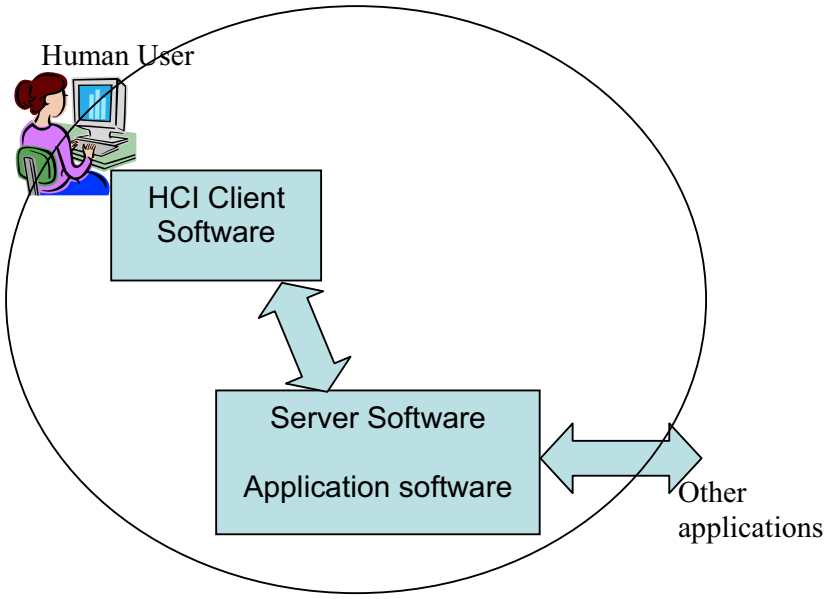


Fig 1. An Application showing major software components

As outlined in the abstract many CFPS counters believe that only a whole ap
sized using the IFPUG method. This is sometimes cited as a reason for usin
of functional sizing. Clients however regularly require of us that we provide
Function Point sizes of infrastructure software separately from other softwar

"Most people agree that function points work perfectly well when measuring business systems. I often find myself, however, explaining why this robust software sizing metric works equally as well when measuring less traditional environments such as middle-ware applications. There are often heated debates on whether International Function Point Users Group (IFPUG) function points can truly represent the "size" of non-business applications. In our client work, we have found that IFPUG function points do accurately represent the size of less traditional environments." Roger Heller Vice president QPMG (www.qpmg.com/middleware.htm).

There are circumstances when it would be desirable to arrive at a functional size of one or more infrastructure software components separately from the application as a whole. Clients need the sizes of infrastructure software in an attempt to understand the range of productivities that they encounter. The reasons that a Function Point count is required for infrastructure software is because most clients have applications made up of different technologies, which may be quite diverse. Clients have a need for this size information to assist them in the management of their software development and maintenance activities (Fig 2). Along with these technologies the development often uses different development teams or indeed components bespoke or otherwise developed by several separate companies.

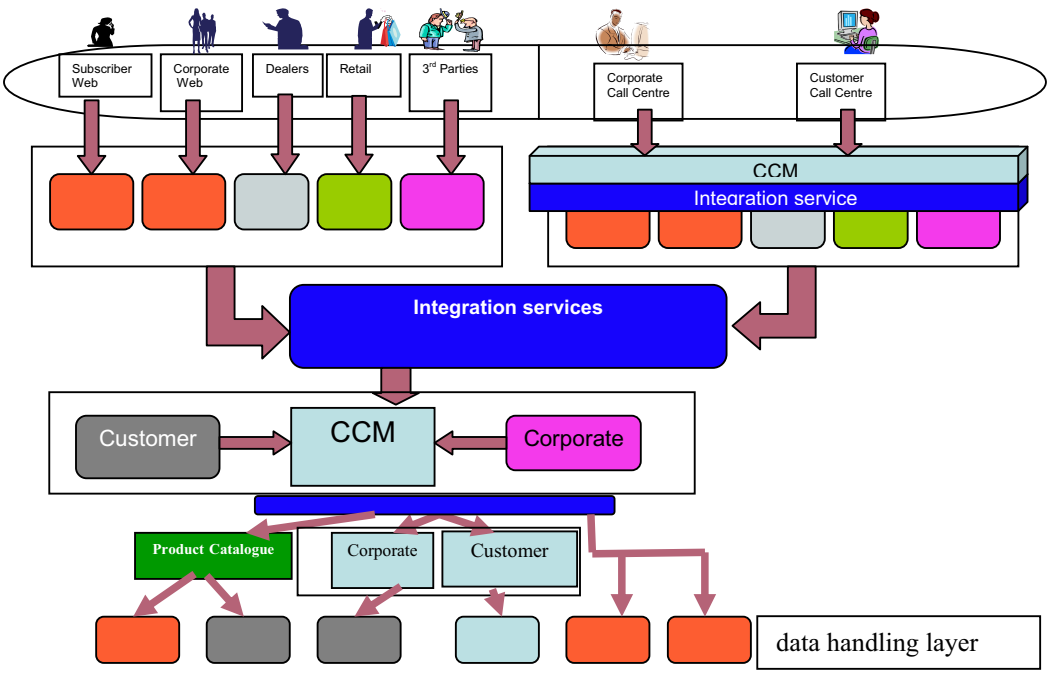


Fig 2. Multiple Technologies, Teams, Suppliers
Different Colours Represent different Suppliers
Languages: COBOL, C++, Java, VB, Business Objects, HTML/XML

Some outsource contracts have Service Level Agreements (SLAs) which are based on the use of Function Points to determine if certain productivity or cost goals are being met. These contracts start by measuring a range of projects to arrive at the baseline data on which the targets are based. The common feature of such baselines is that the figures generated are the result of the analysis of too few projects, and this restricted range does not always reflect the diversity of enhancement environments that may be encountered within a single application. It can also be difficult to separate applications – what were once distinct and clearly separate can become highly integrated and identifying a particular application may be difficult (Fig 2.).

2. Investigation of the rules of IFPUG

The investigation involved a reading of the 4 parts of the FPUG manual in detail to determine if and where the IFPUG rules and guidance would disallow the practice of sizing infrastructure.

Definition of User

The Current IFPUG definition of a user is:

> **User.** Any person that specifies Functional User Requirements and/or any person or thing that communicates or interacts with the software at any time.

As can be seen this allows for any person or thing to be a user so another piece of software is clearly included.

NB the acronym FUR will be used to represent Functional User Requirements in the rest of this document.

Boundaries

The Rules for boundaries are given as:

> The following rules must apply for boundaries:
> - The boundary is determined based on the user's view. The focus is on what the user can understand and describe.
> - The boundary between related applications is based on separate functional areas as seen by the user, not on technical considerations.
> - The initial boundary already established for the application or applications being modified is not influenced by the counting scope.
>
> **Note:** There may be more than one application included in the counting scope. If so, multiple application boundaries would be identified.

These rules tell us that we must determine the boundary based on the user's view, the focus is on what the user can understand and describe, at first glance this presents a problem as a piece of software cannot obviously describe what it perceives as the boundary. However the person who specified the FUR must be able to in order to establish what FUR apply. In practice we find users are often aware of the existence of infrastructure software – they have after all funded its procurement. Thus infrastructure can be seen to be user identifiable in that the user can identify the need for it and the functionality which it provides.

Taking another look we see that users are able to recognise workflow components and they are aware if their functionality and indeed specify maintenance and development activities required to enhance these components in order that the workflows being used are kept up to date with the desired business processes that the workflows drive.

In practice there is not usually a difficulty determining the boundary of an infrastructure software component, these are usually determined by examining the specification for the infrastructure.

However let is examine what is defined as the Users View to see if that provides any difficulty.

Users View

The IFPUG definition of the users view is given below.

> A *user view* represents a formal description of the user's business needs in the user's language. Developers translate the user information into information technology language in order to provide a solution.
>
> A function point count is accomplished using the information in a language that is common to both user(s) and developers.
> A user view:
> - Is a description of the business functions
> - Is approved by the user
> - Can be used to count function points
> - Can vary in physical form:
>
> catalog of transactions,
> proposals,
> requirements document,
> external specifications,
> detailed specifications,
> user handbook

At first glance there might be a difficulty as the users view is approved by the user and a piece of software cannot give such approval – however nor can any other thing so the definition is some what contradictory with the definition of a user. However the specifying person can give approval and the approved FUR are available in the requirements document for the piece of infrastructure under consideration.

Determination of the purpose of the count is based on the business problem that need addressing – sizing infrastructure is likely to meet needs such as determining the size of tasks being carried out by different development teams – infrastructure is usually maintained by a different team to that which maintains the application software.

Scope should present no problems as the scope of the count may include one or more applications and that can equally well include infrastructure software.

IFPUG Counting Rules

The rules for counting data and transactional functions are quite lengthy and the authors do not intend to replicate them here. The reader is referred to the IFPUG CPM Release 4.2.1.

In the case of data functions there is no apparent difficulty infrastructure does indeed maintain files and access files from other applications and hence the counter will be able to determine the ILF and EIF data functions.

Transactional functions such as EI, EQ and EO will exist in the interactions between the infrastructure and the other software components. Nothing in the counting rules for these transactional functions suggests that will be a difficulty in sizing infrastructure.

Remainder of the manual

Part 2 of the IFPUG manual is concerned with the identification and sizing of files, code data and the rules for shared data. These rules should be applied equally to infrastructure as they would to any other piece of software being examined.

Part 2 of the manual is also concerned with the rules for counting enhancements and if an enhancement to an infrastructure application is being counted then these rule will be applied – nothing in them suggests that this will be a difficulty.

The IFPUG Manual Part 3 contains examples of counting situations and the resolution of the examples including the reasoning. There are no examples of the counting of any infrastructure applications. However that does of course no preclude the counting of infrastructure.

The IFPUG manual Part 4 contains some appendices and the Glossary of terms.

Appendix A concerns provides tables to assist the counter in recording and summarizing the counts of function points and the Value Adjustment Factor. These can be applied to infrastructure as with any other application.

Appendix B Gives an overview of the changes between release 4.1.1 and 4.2 the results of an impact assessment of the changes between the releases. None of this has any effect on the counting or otherwise of infrastructure.

Appendix C is the reader request form to facilitate the submission of comments and request for changes to the IFPUG Counting Practices Committee.

Finally part 4 of the manual includes the Glossary of terms – infrastructure is not mentioned in this glossary.

3. Practical investigation

At the request of a client an investigation was carried out to determine if the inclusion of the FP size of a middleware would provide a better fit with the effort expended.

In the case of a few projects the cost based upon the FP size has been shown a considerable variance from the cost based on effort. This is apparently because these projects involve a considerable amount of work on middleware software applications. Middleware has previously not been included in counts, of FP size, because they have not been defined as having a boundary. Middleware is currently considered by the boundary definitions as being within the application boundaries of those applications to which they provide services.

The investigation was carried out in collaboration with Christine Green of EDS. The Authors were tasked to carry out a study to see if:

- Middleware software could be sized using IFPUG FP method
- Middleware could be sized using the COSMIC FP method
- Establish the degree of difference produced by the two methods
- Identify to what extent the difference might suggest the use of an alternative measurement method to IFPUG FPA
- Propose a way forward to assist the resolution of sizing and costing projects with a substantial impact on middleware functionality

The Study

C Green of EDS and A Rollo from SMS worked together on the approach.

First discussion was associated to the definition of the user view and User Recognizable prtions of the definition

Since the Client (representative for the user) were clearly stating in the requirement definition changes needed within the Middleware – as well as changes needed to the Front-end systems such as Billing or CRM the client clearly recognized the existence of the middleware. The Client had even registered the middleware as an application and given it a name.

Next question where associated with the Scope and Purpose of sizing the middleware. In this Case the scope where to size the functionality delivered to the Client of all changes to both middleware and Front end software (customer facing). The purpose was to size the requirements in order to size the requirement that went into an enhancement - added, changed or deleted functionality - of the applications.
The most difficult assessment had to deal with the boundary definition. If the middleware in question where not identified as a separate boundary the sizing would not be possible.

In this discussion it was decided to include the two Function Point methods – Cosmic and IFPUG FP.

The Cosmic FP has introduced two definitions – Boundary and Layers. Due to the Layer definition in Cosmic FP the middleware was definted as a separate boundary. Recognized by the user.

IFPUG FP an assessment needed to be completed in order to clearly identify the Middleware as a boundary.

Definition of the Purpose of the Count	Assessment Result
The purpose of a function point count is to provide an answer to a business problem.	*Changes where to enhance the Billing approach to include alternative billing approach*
Determines the type of function point count and the scope of the required count to obtain the answer to the business problem under investigation	*The Count was an enhancement count and the scope where to size the enhancement to all involved applications*
Influences the positioning of the **boundary** between the software under review and the surrounding software; e.g., if the Personnel Module from the Human Resources System is to be replaced by a package, the users may decide to reposition the boundary and consider the Personnel Module as a separate application	*The Changes identified involved both changes to the Billing Application and a Middleware Application that was used by both the Billing Application and Several other Applications*
Definition of Boundary	**Assessment Result**
The application boundary indicates the border between the software being measured and the user.	*The user is identified as a User or thing that interacts with the Application Boundary. In this case the User is a thing - the Billing Application.*
The application boundary:	
Defines what is external to the application The conceptual interface between the 'internal' application and the 'external' user world	*External to the boundary of the Middleware would be Users interacting directly with the Middleware but using the GUI in the front end Billing Application. As well as Billing Application (the thing) interacting directly with the Middleware.* *Internal for the Application would be Billing and other applications.*
Acts as a 'membrane' through which data processed by transactions (EIs, EOs and EQs) pass into and out from the application	*Data are processed using transactions. Sometimes dta is processed from Billing Directly into Middleware. Some times data is processed using an output transaction from the Billing system transforming or deriving data stored and maintained within the Billing system for usage for other External Boundaries interfacing with the Middleware*
Encloses the logical data maintained by the application (ILFs) Assists in identifying the logical data referenced by but not maintained within this application (EIFs)	*Data are maintained within the Middleware application. Some data is uniq for the Middleware - derived from data maintained in other applications. Some data are maintained and stored in the Middleware even though different interfaces is used for storing data.* *The Middleware sometimes read - sometimes receives data. In cases where data is read EIFs are identified.*
Dependent on the user's external business view of the application. It is independent of technical and/or implementation considerations	*The user view the Middleware from a business perspective in order to eliminate the redundancy in data storage as well as collecting all information in one storage area instead of using reference tables in a many to many relationship…*
Boundary Rules	**Assessment Result**
The following rules must apply for boundaries: The boundary is determined based on the user's view. The focus is on what the user can understand and describe.	*The User understand and describe requirements specifically for the Middleware Application. The user Clearly distinction between requirements for the Front end and requirement for the middleware*
The boundary between related applications is based on separate functional areas as seen by the user, not on technical considerations.	*The middleware are supporting the functional area of some specific data groups as well as the ability to store and maintain data in one area to avoid redundancies and m:m interfaces between different applications.*
The initial boundary already established for the application or applications being modified is not influenced by the counting scope.	*The scope is to size everything. The boundary will not be one boundary - just because the project of enhancing both the billing system and the middleware are done together. If two projects where identified the question of boundary would have been easier - since scope would change - but the identification of the two boundaries stay the same even though the same project plan and team are used.*

Fig 3. Assessment to identify the middleware boundary

Now that the boundary could be defined as being the Middleware application for both Cosmic and IFPUG Method the counting activity could begin.

Christine Green of EDS was to size the middleware component of a project using IFPUG FPA, A Rollo of SMS would size the software using the COSMIC method the counts would be reviewed by the opposite party and discussed to resolve any differences in functionality counted.

It is to be expected that the two methods would produce a different result though experience in other studies has shown that the difference is minimal unless projects are quite large, this would be rare in enhancement projects. The unadjusted IFPUG FPA count is reported as an assessment of the Value adjustment factor (VAF) of the middleware had not been completed. A VAF assessment needs to be carried out across all functionality of an application and the enhancement project under consideration does not provide the necessary information for an assessment. In the case of COSMIC there is no adjustment factor calculation required.

Results

The measurements were conducted and the agreed results are as follows

IFPUG FPA Size is 177 FP
COSMIC CFP size 192 CFP

This amounts to a difference of some 15 FP which is under 8% this is within the generally accepted level of accuracy of these methods (10%).

As a result of these measurements we conclude that there is no significant difference in the measurement value to be obtained by either method.

Biggest issue is the identification of the middleware as a separate boundary. This needs to be done with care. If the middleware has only been introduced as a technical interface between front end and back end, and is not recognized by the user. Not having maintained data that are recoginzed as other than copy or images of the data stored elsewhere it is should not be identified as a logical boundary – but purely a technical implementation. In the study this was not the case.

A detailed analysis of the effect on project costing was to be carried out by the client and the outsourcer. Initial indications are that the addition of the middleware size has accounted for a significant proportion of the variation apparent on the basis of the baseline data.

The Authors concluded that there was no reason why middleware applications should not be sized and recommended that the IFPUG method should be used for middleware as well as for other project sizing. Since the IFPUG Method clearly could be used for sizing middleware under the current CPM definitions and rules. In that case there was no recommendation to change the currently in-use sizing method.

4. Reporting the Size of Infrastructure.

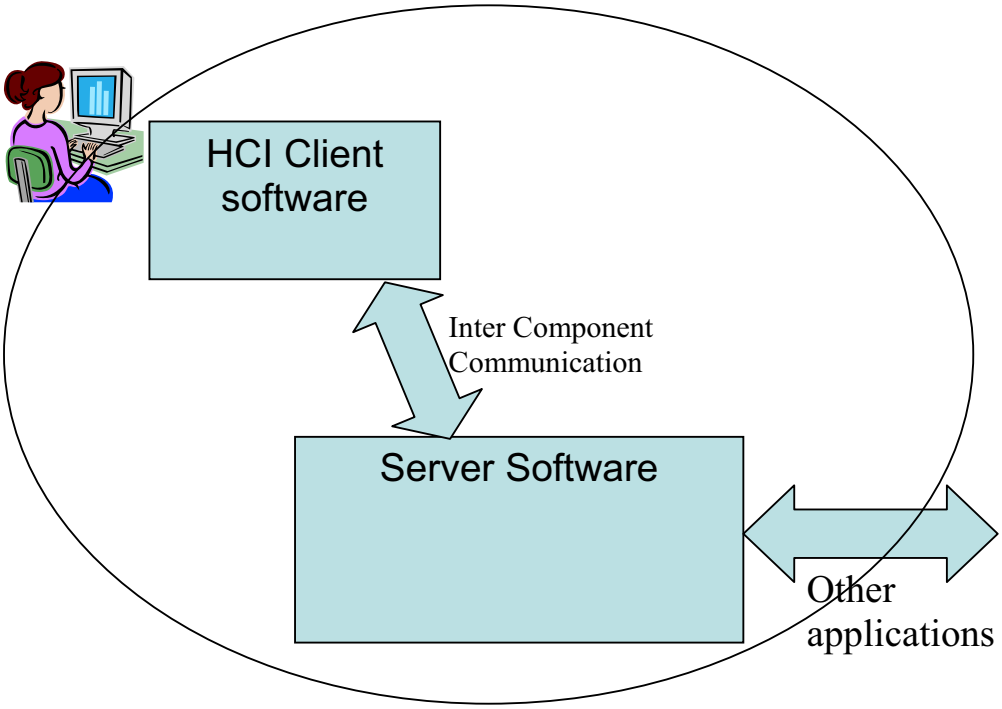
As with any function point count the sizes counted need to be reported to the management that requested the count.

Infrastructure components must be reported separately. It is not valid to simply sum the various components measured and report that as a total size of an application or project. It makes no sense to sum the figures after all the reason for sizing infrastructure software is to gain an understanding of the variance in the development environments then the sizes must be treated separately.

Consider a simple application with components, as in Fig 1. In this case there are two major components which it has been determined need to be sized independently

Infrastructure software sizes should NOT be included in the contribution to size for an application. The sizes of infrastructure may be appropriate in terms of projects dependant upon the purpose of the count. Therefore when reporting the sizes of infrastructure software it is imperative that the sizes are not simply summed and reported as an application size. In Fig 1 (repeated below) the users of the application are a human user and other applications. However when the components are sized separately then this will have included a contribution to the size of both of the major software components made up of the transactions that pass between the components. This contribution will need to be removed before the size of the application can be reported.

NB Remember the application size can only be made up of the contribution of transactions which cross the application boundary.


HCI Client software
Inter Component Communication
Server Software
Other applications