Exploring the Issues of Boundary Definition in the Application of COSMIC-FFP to Embedded Systems

Jacky Keung, Suryaningsih, Ross Jeffery
National ICT Australia &
School of Computer Science and Engineering
The University of New South Wales
Sydney 2052 Australia
{jkeung, rossj}@cse.unsw.edu.au

UNSW-CSE-TR-0336

Abstract

Software sizing plays an essential role in software management and in providing input for estimation and benchmarking purposes. Despite the claim of emerging popularity of function points as a size measure, it is not widely accepted in all software domains. The most popular technique is Function Point Analysis that has become the "de-facto" standard in the business application environment. When applied to non-MIS systems many researchers have criticized the counts as misleading and not reflective of the size of the systems. The release of COSMIC Full Function Point technique is aimed at overcoming these shortcomings. This paper presents a single-case study in a telecommunication company to examine the applicability of the COSMIC Full Function Point technique in the domain of embedded telephone switching systems (a type of real-time system). Through the experience of this study, it is found that there is very limited experience in this area. The current counting convention is thought to be inadequate in many areas such as peer-to-peer sizing and that the field is still evolving. Due to uncertainty and ambiguity in the measurement process, counter's subjectivity plays an important role in function point counting.

Keywords: Software Metrics, Software Cost Estimation, Empirical Software Engineering, case study

1 Introduction

Computer software has become an important part of business operations. Effective management of software development and support is often a difficult challenge. Managers are faced with problems such as budget overruns, late deliveries, and difficulties in justifying the value on investment and poor software quality. These problems can be attributed not only to technical issues but also management issues.

Managers are often unable to setup reasonable expectations for project estimates or benchmarks for project evaluation simply because of the lack of standard metrics in the organization to measure the software project. More often than not they have to rely solely on their experiences and "gut feelings". Therefore, from the management perspective there is a need for more objective and quantifiable information for project estimation and control.

A famous quote in this area by DeMarco states: "You can't control what you can't measure" [25]. Alan Albrecht introduced the concept of functional size measure using function points in 1979, with the vision to "develop a relative measure of function value delivered to the user that was independent of the particular technology approach used" [7]. However, several researchers have observed, the application of function point technique is not as straightforward or simple as it first seems. It requires experience and expertise [8][9][10][24]. The establishment of the measurement context to be used during the measurement process requires some subjective judgments to be made. This is in turn could influence the final counts generated. Though the counts may differ they are not necessarily "incorrect" counts according to the rules of the counting manual.

According to Desharnais [3], though this might seem to be an unscientific approach, this is the current practice in the industry. As another researcher Rule [20] points out, function point based metrics are considered as giving a relative measure of size rather than an absolute measure. Abran [2] also suggests that it is possible to get different answers out of a measurement process when the measurement instrument is used to measure the same context but for different purposes or

perspective. However, he also emphasizes that "you get different answers depending what you want to know, and not depending on the measurement instrument".

Its descendant is the Full Function Point (FFP) technique; it is an extension of the traditional Function Point Analysis (FPA) specifically developed to cater for both MIS and real-time system measurement [17]. COSMIC (Common Software Measurement International Consortium) is a group established by 6 major countries in the world, with the aim to achieve an international standard set of software measurements that is ISO compliant. The COSMIC Full Function Point method is a refinement of FFP, Mark II and the FPA technique, to address a variety of software domains especially MIS and real-time systems [23]. It is aimed impact to resolve the different interpretation of the original FPA concepts, which resulted in counting inconsistencies. [6]

Hasting [5] states that formal model validation is essential for the credibility and adoption of a technique. He notes that "none of the models would seem to have undergone rigorous formal verification. There is only limited evidence published on empirical validation for the function point, feature point and MKII models."

For the industry to progress and to be able to use these metrics in their own organization as well as to benchmark against another, raises the need for international standardization on methods and ways to convert one to another (Total Metrics 1998).

This study focuses on two issues that were raised by researchers based on observation and experience during the application of the function point technique. Firstly, an important issue that has been raised by researchers is the issue of the counting boundary. The ISO/IEC 19761:2003 defined boundary as "A conceptual interface between the software under study and its users". The "appropriate" definition for the counting purpose is essential since the boundary establishes the area to assist the counters to decide without ambiguity what should be included and excluded in the counting process. According to Desharnais [3], "Changing the application boundary will change the functional size". Secondly, in relation to the possibility of having different size measures for a system which are dependent on the point of view of the describer. Hasting [5] recognizes that software could be measured in terms of system size which refers to the measure of functionality delivered to the business users or the component functionality which is a measure of the functionality delivered by sub-systems. Hasting [5] also recognizes that the FPA based models can not be guaranteed to be partitioned into independent components, which means that the sum of the components functionality might not be equal to the total system size.

The impacts of the two issues described above on the FFP counts are explored via the measurement of an enhancement project of a telephone switching system (real-time system) from a leading telecommunication company. Three system models are proposed presenting two levels of system abstractions and two boundary views. Details of findings are presented and discussed in section 4 and 5 respectively, and section 6 concludes the paper.

2 Research Design

As argued previously, there is no clear definition of what constitutes the "right size". There is the possibility that the same systems could have different size measures that are compatible with the counting rules. The measure as mentioned previously is dependent upon the point of view of the describer. The different perspective that could arise with the different interaction experienced or level of details concerned necessary. This issue involves broader concerns in relation to the

usefulness of one perspective to another and that often one size measure is not adequate to represent a system size.

There are five major steps in calculating function points. They are:

1. Determine Type of Counts
2. Identify counting boundary
3. Determine Unadjusted Function Point Count
4. Determine Value Adjustment Factor (VAF)
5. Calculate Final Adjusted Function Point Count

The Value Adjustment Factor is not used in this case study, due to criticisms made by researchers [7][10][12] that the adjustment process is not effective in improving effort prediction. Lokan and Abran [13] also suggest that the FFP technique assumes that general system characteristics (GSC) and Value Adjustment Factor (VAF) do not add value to functional software size. Therefore this study will only use the first three steps (i.e. 1, 2 and 3), in order to generate the unadjusted function point counts.

The approach taken to explore the impact of the subjectivity issues on the function point counts using FFP technique is therefore:

1. Determine Type of Counts: this establishes the purpose and scope of the counting, and defines what needs to be measured.

2. Identify counting boundary: There are two phases involved in identifying the counting boundary. As mentioned earlier, the counting boundary is dependent upon the viewpoint of the describer (system vs. component sizing). This involves the separation of the application to be measured to its peers (within and different layers) as seen from diagram below (Figure 2.1).

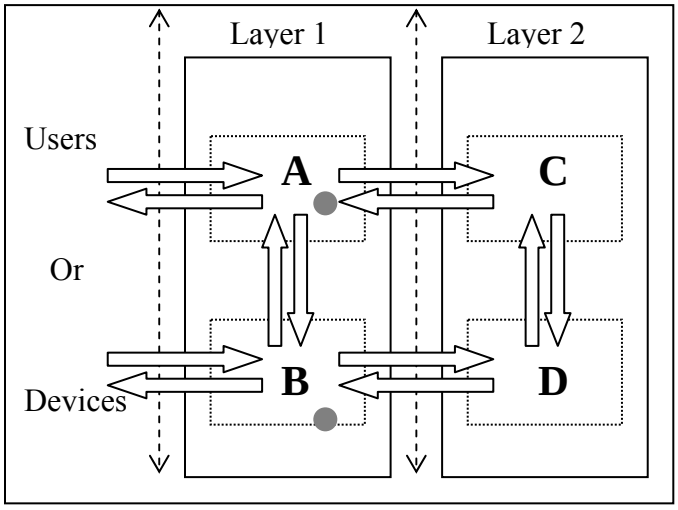

Figure 2.1 Possible System Interactions

Figure 2.1 above summarizing the possible interaction any system might have (the grey area/dot represents changes made to the applications). Layer 1 applications refer to applications that support the end-users (human or devices), while layer 2 applications refer to infrastructure software.

This is then followed by the assignment of boundary for counting purpose (figure 2.2). The first (boundary 1) refers to design from the top-level design perspective, which is looking at the system size as one large system with the boundary drawn between the system and the end-users.

The second level refers to the explosion of the system size into finer granularity increasingly dependent upon the system design, which is taking account of all the sub systems associated with the performance of system functionality. The boundary is to be drawn between sub-systems in which the users of the sub-systems will no longer be the direct end-users but the various sub-systems (boundary 2).

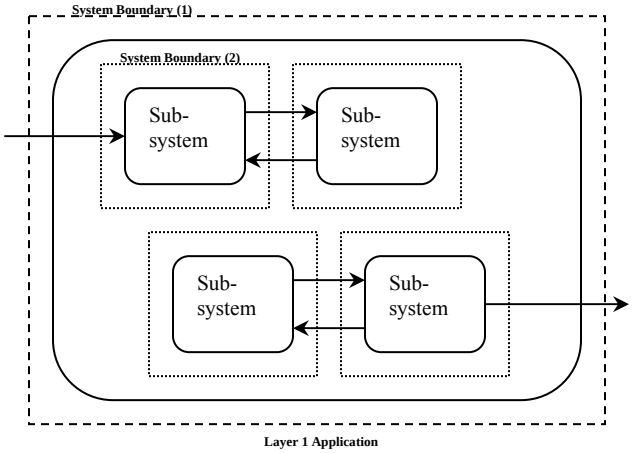

Figure 2.2 Assigning Boundary

3. The assignment of the function points follows the pre-defined context as established by the boundary as well as applying the rules and guidelines provided in the counting manual.

Finally, comparison is made on the counts generated.

3 Methodology

The organization under study is a telecommunication company in Australia involving mainly the switching division. Their switching division is responsible for delivering and maintaining switching systems for the provision of national and international telecommunications services to enterprises and the general public. Approximately 4 million lines of code are maintained to provide these services. Throughout the years, many projects were initiated to accommodate changing client demands for customized services or call facilities. The need to be able to selectively forward calls to pre-defined destination resulted in the Selective Call Forwarding (SCF) enhancement project, which then becomes the project under study. While it is recognized that a single-case design choice may reduce the external validity of the study, due to the exploratory nature of the study, it is of greater interest to thoroughly explore a single case rather than to obtain broader information generalization. According to Stake [21], this type of research is classified as an instrumental case

study approach. This is where a particular case is examined to provide insight into an issue and to illustrate how the concerns of the researcher are manifest in the case.

The use of a completed enhancement project (SCF project) is to illustrate the issues to be investigated. In an enhancement project, software development involves mainly the modification of the base application to perform additional or new functions. Most of the organization's project development involves enhancements to the existing application rather than purely new development.

3.1 Data Collection

There are two stages involved in the data collection. The first stage in the counting process is the establishment of the purpose of the counting and the measurement context (boundary and level of abstraction) to be used in the counting process. This is where a need for an understanding for the overall system is necessary. To achieve this, interviews were used. This is to maintain the flexibility to discover or unfold the participant's view [15][25]. The participant is given freedom to express their perspective; the involvement of both the system analyst and designer in the study also means that better understanding of the system can be achieved.

The second stage is where the detailed data needed for to perform the FFP count is gathered from the requirement specification. Again, this is with the help of the system experts.

The generation of the unadjusted function point counts requires detailed information of the SCF project in terms of the data element used, and inflow and outflow of data. Documents that are needed to facilitate the counting would be those system documentation that show data used in the system as well as the inflow and outflow of these data to and from the application being modified and also the sub-processes that make up those processes. In performing the function point calculation not all of the documentations are used. In this case, only the strategic document, requirement change request (RCR) and the Top Level Design (TLD) are used.

Strategic Documentation describes the new requirements to be developed as well as the top-level overview of the implementation. Requirement Change Request (RCR) outlines the new service specification that the client demanded. Top Level Design (TLD) outlines the changes (additional, modification and deletion) that are needed on the infrastructure (identified from the Technical Requirement Specification) in order to achieve the specified requirements.

It is not unusual to use triangulation to validate scientific findings; the use of multiple sources of evidence provides in-depth understanding of the phenomenon being studied. [26][21][4][11] The data collected from both system documentation and interviews is essential to reduce the likelihood of misinterpretation and bias of the researcher.

Another strategy adopted to minimize the potential researcher bias as well as to ensure the completeness of the findings reported was the Member checking technique. Member checking refers to the process of having the system experts previously interviewed to examine the draft of raw data collected [11]. By doing this, accuracy and completeness of the data collected is ensured through the revision, correction or additional feedback received. This strategy is also important remembering the complexity that is involved in studying a system within a short period of time. This process is also considered as a way to establish the chain of evidence thus strengthening the validity of the study.

4 Results and Findings

Identifying what is to be measured is the first stage of the development of measurement context. The purpose of this measurement process is to calculate the function points associated with the added, deleted or modified functionality introduced by the implementation of Selective Call Forwarding (SCF) enhancement project. It is therefore necessary to firstly identify the base package associated with the project and to understand the functionality that the base package offers.

A scenario below is presented to describe the basic call operations.

1. A call operation is triggered when the network receives an off-hook signal from a subscriber (Subscriber A picks up his/her phone).
2. The call control process then has the duty to identify the subscriber and check his/her profile to indicate that the subscriber has permission to make a call.
3. If the subscriber has permission to make a call, the network then gives a response signal in the form of dial tone.
4. The subscriber then responds by pressing digits on the phone keypad, i.e. entering the call destination number (Subscriber B).
5. Upon receiving the digits, the call control process analyses the digits to identify the appropriate action to be taken.
6. The call control in a basic call operation then checks the destination profile making sure that the destination exists and identifies the possible routing direction.
7. The network then checks the line availability of the destination party.
8. If the line is busy the network gives the caller a “busy tone” otherwise a “ringing tone”.
9. A connection between the two parties will be established only if the destination party (Subscriber B) answers the phone.
10. Once both parties are connected, the main control process will no longer be in charge of the operation. The charging and measurement application then takes over the process to collect the data necessary for call billing and accounting purposes.
11. The call control is in charge of the operation when either party terminates the call.

4.1 The Selective Call Forward Enhancement Project

Selective Call Forwarding (SCF) is an incoming call management feature, which allows the served user to automatically and selectively forwarded incoming calls. For this purpose the served user can define a screening list that contains a series of calling numbers on one hand and unique destination (forwarded-to) number on the other hand.

Based on the previous example, Subscriber B has SCF facility offered and activated and has specified the screening list below for this facility (Figure 4.1). When subscriber A calls subscriber B, subscriber A’s call will then be directed to the destination subscriber C. However, if subscriber D called subscriber B then the call will remain directed to subscriber B since the number is not listed to be forwarded. The diagram below shows such interaction as an example.

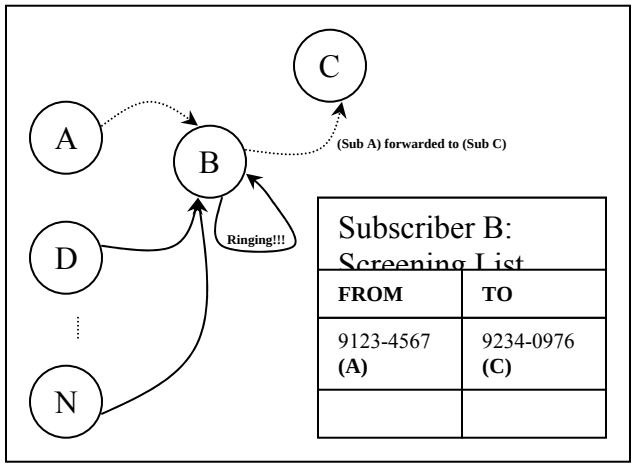

Figure 4.1 Selective Call Forwarding

In order to perform the call operation scenario above, the company has to maintain a combination of hardware and software. In this case, we are only interested in the software implementation. We have discussed the scope of the measurement that defines what needs to be measured. However, it does not specify how the measurement should be carried out. Recall in our design that identifying the counting boundary is the second step of the measurement process. Function point based metrics are dependent upon the concept of boundary since the rules and definitions use the boundary to identify both data transaction function types.

The diagram (Figure 2.1) represents the interactions that the switching system could have with the end-user (human or device) and the interaction between software layers or system components. This supports the argument on the issue that system functionality can be measured in terms of system's overall functionality delivered to the end-user (human or devices) or systems components functionality. [5]

System sizing represents the overall system functionality received by the end-user that is independent of the technology used. This is useful for external organization evaluation purposes, e.g. in communicating functionality delivered to client.
However, this type of sizing has very little value for internal evaluation purposes. This is because often system development involves components that do not contribute to end-user functionality (e.g. change in infrastructure software) or when the development of a large system is partitioned into sub-systems that are separately managed by different teams. In this case the above view cannot be used to compare the different teams' productivity. Therefore the approach here is to size system from both views.

The first step of system sizing is to identify all the major groups of data movement between the boundary of the measured software and:

- Its human user operators
- The boundaries of other software
- Or engineered devices

The scenario discussing the basic call operation is based solely on the interaction of the subscriber with the call operation application. Based on this scenario, we can identify that a normal user of the system (call subscriber) interacts with the system via engineered devices attached to the network which translate the user input so that can be understood by the system and vice versa. User input can be of three forms, the first input is initiated when the user picks up the telephone, and a signal is then sent to the system indicating that someone is trying to make a phone call, triggering the main call control process. The second data entry occurs when the user dials the destination digits and the third possible data entry is in the form of incoming calls. An incoming call is a data entry that is sourced from the trunk of the network, which could possibly come from overseas.

The output of the system will be mostly in the form of "dial tone" or "ringing or busy tone" given back to the user to the appropriate response in the form of recorded voice announcement such as e.g. "Service is active" during a successful activation.

However, it is also possible to have direct human interaction with the system, i.e. via the operator. The selective call forwarding feature as explained previously is given on a subscription basis, thus the operator should have the capabilities to setup specific features for a given subscriber by updating the subscriber's profile.
At the end of this step the interaction with hardware and operators are known. The next step is to identify the applications that handle the interaction. Application software is defined as software that delivers functionality to support the organization's core business [18]. There are two broad categories of applications that are relevant in this case, i.e. The application that supports the call operation and the application that helps in maintaining user profile for the business (used by operator only) the diagram below illustrates this scenario. (Figure 4.2)

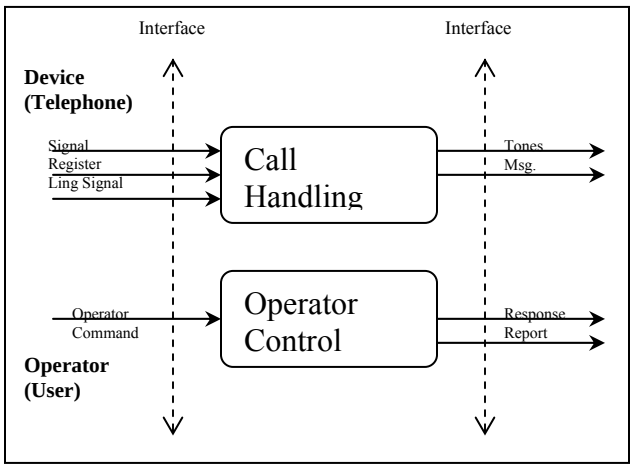

Figure 4.2: Operator, Device and Application Interactions

Functionality delivered by system components can be further broken down into functionality delivered between the software layer and the functionality delivered by the sub-systems. A switching system consists of the application software, kernel and operating system. A simplified three-layer model approach for the discussion of software layers in this case is thought to be adequate since the SCF project studied involved only the modification of the applications layer. Each of these layers works together in performing a call operation.

The application software is supported with distributed database and a multi-tasking operating system. The kernel and operating system are acting as supporting software that supports the application software in performing its function. For example, the kernel assists the application software in updating the database. Towards the hardware side, the device driver is embedded in the hardware to act as an interface between the external users of the system, for the purpose of converting system output so that it can be understood by the subscriber, e.g. giving different tones or ring types as a result of an operation.

Looking at the lower level sub-system view, the call handling application is very large that can be further examined and broken down into various sub-system components which have their own unique functional contribution to the overall call operation. We could identify the call handling application that consists of two broad categories. The "heart" of the system (the main control process – CFCS) governs all the call operations, and the "supplementary services" which provide specific services as requested by the "heart of the system".

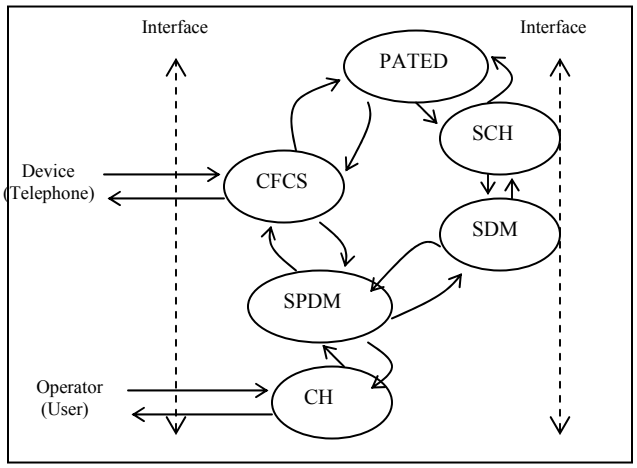

Figure 4.3: Component Interactions

The main control processes in the case of a call operation are:

1. Call Facility Control System (CFCS – main control process). The heart of the system that governs the overall call operation. Upon receiving a trigger from the subscriber (off-hook), this process has the duty to determine the action needed to satisfy / fulfill the request.
2. PATED (Prefix Analyzer) – This process has the duty to analyze digits and identifying the specific association, e.g. a normal call (local, international) and subscriber control code (facility activation/deactivation).
3. Semi-Permanent Data Manager (SPDM/LSIF) – maintains the subscriber information (i.e. subscriber profile)

4. Subscriber Command Handler (SCH) – Handles and control subscriber's activities, and allows the subscriber to have control over his/her profile e.g. activate a certain facility.
5. Subscriber Data Manager (SDM) - This process changes the subscriber's data based on subscriber activity e.g. update the subscriber profile on activation.

Unlike the call handling application that can be further broken down into sub-systems, the management application used by the operator consists of only the Command Handler (CH), which allows the operator to manage the subscriber feature via the interface provided by the man-machine command.

To data peer-to-peer sizing such as illustrated above seems to be an issue that has not been resolved. The recently released COSMIC counting manual (COSMIC-FFP ISO/IEC 19761:2003) recognized that distributed systems are often characterized with peer-to-peer communications. However, the manual offers no explanation on how to size this type of interaction.

4.2 Identify the counting boundary (Assigning Boundary)

The purpose of a boundary is to assist the counter in determining things that should be included or excluded from the counting process. Desharnais [3] pointed out that in developing the counting boundary in practice, counters look at what is managed or consistent with the organizational and budgetary boundaries.

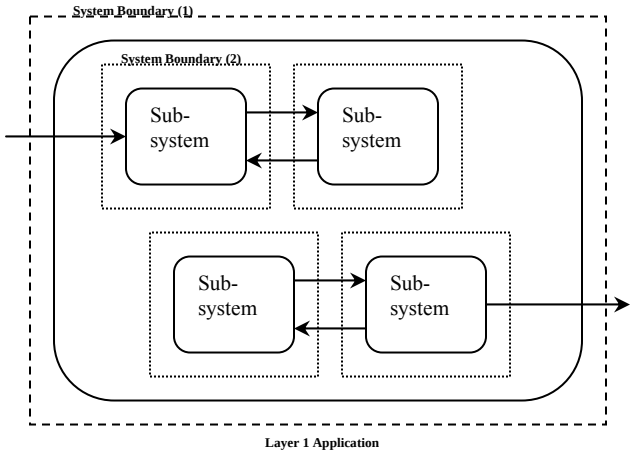

Figure 4.4: Assigning Boundary

Here we proposed and defined two system boundaries, boundary (1) aims at measuring system functionality delivered to the end-user (human or device), whilst boundary (2) aims at measuring system component with each of the sub-system becoming the user of each other.

4.3 The Implementation

The aim of this study is to explore the impact of the issue of boundary and the level of abstraction used to size the system, i.e. to identify whether:

- The different boundary assignment will have an impact on the final counts generated, because the rule of the technique reward function points based upon the definition of boundary.
- The different system view (system vs. component view) will have an impact on the counts generated.

Based on the arguments presented above, two boundaries and two system views are represented by three models:

- Boundary (1) Top Level View (System View)
- Boundary (1) Low Level View (Component View)
- Boundary (2) Low Level View (Component View)

Boundary (1) represents the end-user perspective (human or device) and Boundary (2) represents the sub-system perspective. The system view represents the system overall functionality and component view represents the component or sub-system functionality.

4.3.1 Boundary (1) Top Level View (System View)

This is where we size the system with a boundary drawn from the user perspective (Human or device). From the top-level view, the switching system consists of two applications; the application that relates to the handling of the call operation and the application that used to maintain the subscriber profile for the business purposes. The boundary proposed here is to incorporate both applications as one boundary instead of having two separate boundaries for the two. This is because the operator control application cannot exist independently. Looking at how the system component works (Figure 4.2), we could identify that the command handler which represents the operator control actually reuses the call handling component to update a group of data (i.e. the subscriber profile). Thus we consider both applications as having one boundary to represent the switching system functionality delivered to end-user. Sizing both applications independently will have a larger impact due to the double counting of the same group of data used.

4.3.2 Boundary (1) Low Level View (Component View)

The boundary (1) Low Level View case is similar to the case above; the difference here is that this case represents the explosion of two applications into sub-systems. The boundary drawn here incorporates all the inter-communication between the sub-systems. This is to explore the impact of the boundary definition upon the generation of function points and the impacts of transaction function point type that measures at sub-process level. (Figure 4.4)

4.3.3 Boundary (2) Component Level

The Boundary (2) Component Level here takes the view that the user of the system is now the sub-systems.

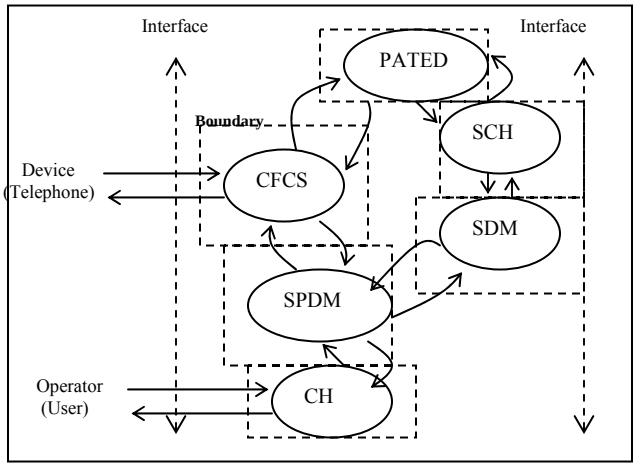

Figure 4.5: Boundary (2) Component Level View

The second boundary is proposed to take into account the data-flow that occurs between the internal processes. In addition, with the argument that the organization in this study allocates their development or maintenance effort and budget at component level and that often the development of the different components are allocated to different groups of developers. Thus this level of abstraction may provide a comparative study between groups, boundary for the call handling application is drawn between the sub-systems.

4.4 Determine the Unadjusted Function Points

According to the FPA technique, an enhancement project function point is calculated by (IFPUG 1994):

Enhancement FP = [(Add + Change + Conversion) x VAFA] + (Deleted x VAFB)

Note: The application value adjust factor (VAFA and VAFB) is not used in this case.

Recent released COSMIC-FFP technique (v2.2) calculates an enhancement project by (COSMIC 2003):

Size(change(layeri /functional processi)) = Σ size(added data movement) + Σ size (modified data movement) + Σ size (deleted data movement)

In this case we adopt the COSMIC-FFP v2.2 for the calculation of the transactional function type since both FFP version use the concept of data-movement or sub-process in the calculation; while the changes in the function point type follows the FPA rules.

4.4.1 Boundary (1) Top-Level View

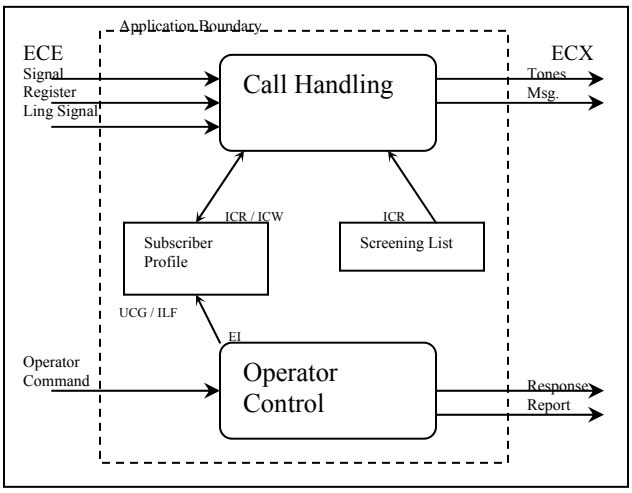

Figure 4.6: Boundary (1) Top-Level View Counting

We identified two groups of data necessary to perform the entire operation; screening list and subscriber profile (Figure 4.6).

During the service invocation, the system reads the screening list to identify calls to be forwarded. It is assumed here that the data structure of the screening list is not available therefore it needs to be developed. The screening list as part of the effort associated with the development of this feature is included, since this facility operation relies heavily on the screening list data. The new group of data (screening list) will consists of the directory number and the forwarded to number.

Screening list data is an updated group of data since the data can be updated by the application during registration of the number to be forwarded and destination number by the subscriber via the screening list editing (SLE) facility.

It is a multi-occurrence group of data because there are many instances of the data for different subscribers. The SCF screening list has the maximum of 30 entries, with minimum space allocated is 5 entries, then an increment of 5.

Since the data is maintained within the counting boundary, the Internal Logical File (ILF) rules of the FPA's should be applied. Data functionality is measured by the number of data element types (DETs) and record element types (RETs) present in a group of data. Count a DET for each user recognizable, non-recursive field on ILF. Count a RET in any user recognizable sub-group of data element within an ILF.

There are 2 DETs in this case: Directory number and Forwarded to number. Because there is no subgroup in this case the screening list is then assigned with 1 RET. Therefore the screening list is considered as having low complexity (2DETs, 1 RET), which is equal to 7 points according to the complexity matrix described in IFPUG 4.1 counting manual.

For activation and deactivation of the service, there would be a new development group of data (call type) to accommodate the different types of calls that the subscriber is entitled to including SCF facility as well as the activate status for the service. This group of data is also used by previous requirement (invocation of the facility) to check whether the facility is active for a subscriber. This group of data thought to be both control and management data because during the subscription time the operator updates this group of data in order to set the service allowance for the subscriber. This has no impact on the point allocation since multiple occurrence type of data uses the same formula to the FPA's.

There are 2 DETs: Service Type and Status. This group of data has no sub-groups and will be assigned 1 RET. Again this group of data is classified as low complexity (2 DETs, 1 RET) and has 7 points assigned.

Function Description	**Function Type**	**FTR/ RET**	**DET**	**FP**
Control data function type:				
Screening List	UCG	1	2	7
Subscriber Profile	UCG/ILF	1	2	7
Unadjusted Function Point Counts:				**14**

Table 4.1: Total UFPC – Control Data Function Type

Transaction function types represent the functionality provided to the user for the processing of data by an application. Therefore, to identify transactional function types it is necessary to identify the processes of the application first.

The occurrence of a transaction (an instance of process) is dependent upon the occurrence of an external trigger. A trigger is an event that initiates a process from a functional perspective. The event comes from outside the application boundary. When an event occurs, data usually enters the software boundary.

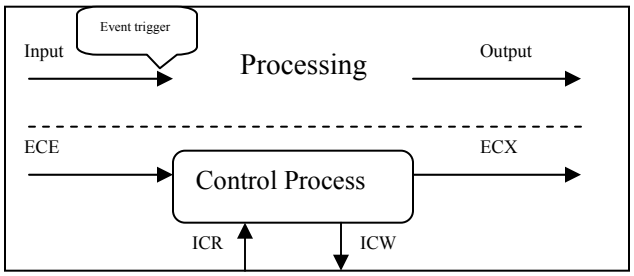

Figure 4.7: Control Process Trigger

From the top-level view, there are only two processes that need to be counted (processes with direct external triggers see Figure 4.7 above)

Process 1. Call Handling

The transaction functionality is calculated through the identification of sub-processes associated with a process. Sub-process is the lowest level of decomposition of a process action on one group of data. It is identified as entry, exit, write and read. The identification of the new added sub-processes followed the requirements from the SCF enhancement project.

Requirement 1: service invocation

- Reading the terminating subscriber profile to check whether SCF service is part of the subscription. This involves the reading of the selective service relation. The service allowance flag and the activate status comprises 2 DETS for data read, that is equivalent to 1 function point
- When the SCF feature is active, read the screening list. The data involved in this case is the screening number, which has 2 DETs, or 1 function point.
- Compare the screening list to the calling line identification. This sub-process does not receive, send, write or read control data; therefore contributed zero function point.
- If a match is found then the CFU facility takes over the process. SCF forwarding feature is inherited from the CFU service. Otherwise the basic normal call takes over the operation. The call will be given normal termination.

Requirement 2: Activation/Deactivation Service

- Reading the terminating subscriber profile to check whether SCF service is part of the subscription. The reading of the data in this case, although similar to the one in the service invocation, it will still be counted since the reading of the subscriber profile in this case represents different functionality.
- Process to update the subscriber active/non-active status if the subscriber has the feature, otherwise returns error message to the caller. This involves the updating of the active/non-active status, which contributed to 1 DET.

Requirement 3: The editing of the scanning list is not part of the effort for this facility, therefore is not accounted for.

Process 2: Operator Control

Requirement 4: The interaction between the operator and the subscriber profile

- Upon the request for subscription for this facility, the subscriber profile needs to be updated. The data involved in this update is the allowance for the service in the subscriber profile. Therefore is equivalent to 1 DET and 1 FTR. This contributes to low complexity thus generates 3 function points.

This fits the definition of External Input (FPA technique): an external input process data or control information that comes from outside the application's boundary and maintained an ILF (IFPUG 1994).

Function Description	Function Type	FTR/ RET	DET	FP
Control Transaction function type				
Call handling – Invocation				
Read Subscriber profile	ICR	N/A	2	1
Read Screening List	ICR	N/A	2	1
Call Handling -Activation/Deactivation				
Read Subscriber profile	ICR	N/A	2	1
Update Subscriber profile	ICW	N/A	1	1
Call Handling process points				4
Operator control				
Update Subscriber Profile	EI	1	1	3
Operator Control Points				3
Unadjusted Function Point Counts:				7

Table 4.2: Total UFPC – Transaction Function Type

The total unadjusted function point counts is therefore
14 + 7 = 21 Function Points. The question on the compatibility of adding results from both methods, i.e. the result of the FFP technique to of the IFPUG is justified by the fact that both techniques are based on the same concepts and measurement process [19].

4.4.2 Boundary (1) Component-Level View

The Function Point generated both from top-level view (system view) and lower level view (component view) using the same boundary should generate the same result. This is caused by several reasons. Firstly, the COSMIC FFP counting manual states that there is a possibility for processes to have multiple triggers: processes may have more than one trigger (one primary initial control event and secondary events). In this case, the entire process is considered as one control process for counting purpose. The primary control trigger in a call operation would be the off-hook signal received by the main control process (CFCS), which then stimulates other triggers to activate other processes such as PATED and LSIF. It is therefore that all the internal control processes would be considered as one control process (call handling) as shown in the previous case. St-Pierre [3] also suggests that data-flow and triggers between internal processes should be ignored because they are considered as mainly dependent upon the system design rather than functionality.

Secondly, although there are numerous data-flow passes between each of the sub-systems (CFCS and LSIF), they are not counted for as entry or exits in this case, because the rules of the technique assign points based on the interaction that occurs with the definition of boundary, that is only data entering and going outside the application boundary is relevant.

In addition the COSMIC FFP technique calculates transaction function types at sub-processes level (the smallest processing step identifiable from a functional perspective), these are including entry, exit, read and write.

4.4.3 Boundary (2) Component-Level System view

In the previous two cases, the data flow between internal processes has not been accounted for due to the use of end-user perspective. In this case, the overall function point counts contribution from the data function type would be the same as in all the previous cases, since the same group of data are involved in implementation of the project. It is difficult to allocate these groups of data to a specific component as both the SDM and SPDM have the capability to update the same group of data, i.e. the subscriber profile. This creating ambiguity on whether the development of subscriber profile should be allocated to SDM or SPDM.

The two diagrams figure 4.7 and figure 4.8 below illustrate the specific components of the switching system invoked during a call operation involving the selective call forwarding facility, that are invocation and activation/deactivation of the service.

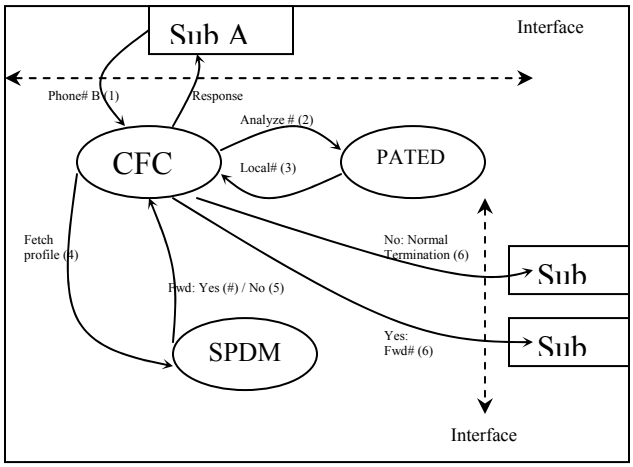

Figure 4.8: The Invocation of Service

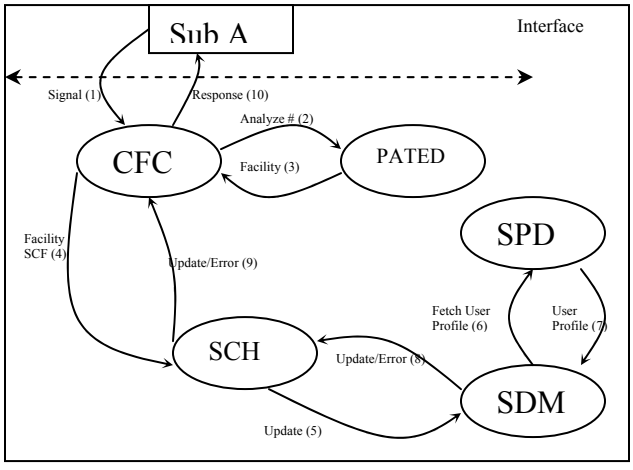

Figure 4.9: The Activation/Deactivation of Service

Impacts on each component:

1. CFCS: There is no change in the CFCS control process; the SCF facility inherits the call forwarding features from the CFU facility. i.e. Once there is a match then the CFU flag is activated. Thus the rest of the call operation is considered as CFU facility (taken over).

2. LSIF/SPDM
Requirement 1: Service Invocation

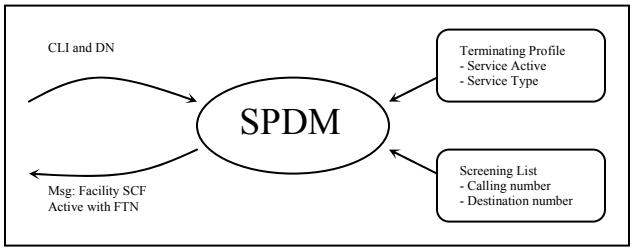

Figure 4.10: SPDM – Service Invocation

There is no change in the data entering this process, the calling number (CLI) and the directory number (DN) representing the caller subscriber as shown above is different to this facility. These two numbers are used in a normal call operation to determine the caller and destination number.

The change in this control process is therefore in relation to the reading of the service allowance and active status – 2 DETs; according to the COSMIC counting manual this change contributes 1 function point.

If the SCF facility needs to be invoked, the reading of screening list is then required. This contributes 1 function point.

Requirement 2: Service Activation / Deactivation

The only added functionality made to this process is in relation to the subscriber control facility offered that allows the subscriber to activate or deactivate the SCF facility (Figure 4.11). Depending upon the request received (activation or deactivation) this control process updates the services status in subscriber profile. The updating of the data on activation contributes 1 DET, equivalent to 1 FP.

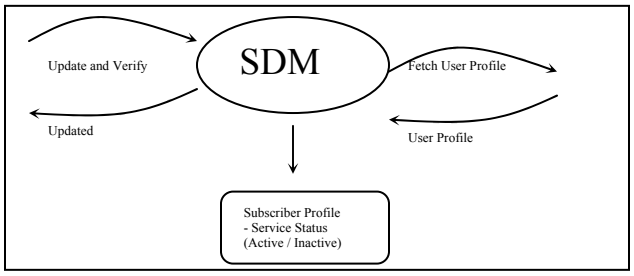

Figure 4.11: SDM – Activation / Deactivation

Function Description	Function Type	FTR/ RET	DET	FP
Control transactional function type				
CFCS	-	-	-	-
LSIF/SPDM				
- Read subscriber profile	ICR	N/A	2	1
- Read Screen list	ICR	N/A	2	1
- Reading of service status	ICR	N/A	2	1
SCH	-	-	-	-
SDM – update subscriber profile	ICW	N/A	1	1
Call handling process points				4
Command Handler	EI	1	1	3
Update the subscriber profile				3
Operator control points				
Unadjusted Function Point Counts				7

Table 4.3: Total UFPC – Component Level

The total unadjusted function point count in this case is the same as the previous counting. This is due to the reason that there is no change in the ECE and ECX between the internal processes, therefore the impact of the boundary cannot be observed here.

The usefulness of the decomposition in this case is the allocation of function points associated with the effort made by each component. Therefore it can be used for estimation purposes as well as comparison between the components or groups/team assigned.

We have calculated the total unadjusted function point counts of an enhancement project from three different models that represents different counting perspective. In the next section, we discuss the results and their implications.

5 Discussions

5.1 The characteristics of the FFP techniques

As mentioned previously, the FFP technique relies on the concept of boundary to identify both data and transaction function point types for the assignment of function points to a system. This has an impact in the sense that the different system view (system vs. component view) will not cause any variation on the counts generated unless accompanied by the different boundary definitions. That is, from the two cases Boundary (1) Top Level View and Boundary (1) Component View, these two models will always generate the same function point counts. This is because the assignment of function point counts depends solely on the concept of boundary. Although there are many in-flows and out-flows of data between internal processes as can be seen from the second case (Boundary 1 Component View), they cannot be accounted for because boundary 1 is drawn to incorporate the intercommunication between these processes.

The introduction of the new control function types seems to improve the measurement ability of the model in measuring control-related functionality. It was observed that a control process (call handling) did require a lot of internal processing in serving a call operation. This process has a

significant number of sub-processes present and that any call operation will not leave the call handling application in a consistent state until all the sub-processes have been carried out. This would represent a single elementary process in the FPA method, which again will be difficult to classify the transaction into input, inquiry or output. In particular, the "Internal Logical File" concept is one of the most difficult concepts to interpret in practice [22]. The COSMIC FFP v.2.2 has a well defined concept of its "Data movement", therefore the new transaction types at sub-process/data-movement level removes this ambiguity.

Another advantage of analyzing the impact of an enhancement project at a lower level of granularity is that any change can be accounted for appropriately, i.e. as [18] pointed out that "only part of the process (identified by sub-process or data movements) is credited for the change". This was again observed in the case study, that it would be easier to take into account the change e.g. an additional sub-process or data movement that the control process has to perform in relation to the reading of screening list, which was previously not necessary.

5.2 The characteristics of the System studied

The characteristics of the system studied will also have an impact on the function point generated. The system under study is partitioned to group common process in which the different components have specific roles of reading, writing, or manipulating data. Because the FFP technique measures at this level of detail (i.e. size (entries) + size (exits) + size (reads) + size (writes)), changes to any of the components can be captured by the technique. However, changes that did not result in the entry, exit, read or write cannot be accommodating by the technique. For example, changes required in the program code involving only the internal algorithm for the purpose of setting service priority cannot be accounted for. This might be due to one weakness of the model as discovered during the field test that 2 out of 81 processes expected to be counted were not detected because they involved only internal algorithm [17].

5.3 The sense of coverage

The application of the FFP technique to an enhancement project showed that the introduction of the new transaction function types (i.e. the measurement functionality at sub-process level) seems to improve the measurement ability of the model in measuring control-related functionality. Measuring at this lower level of granularity removes the ambiguity in applying the concept of elementary process where a large number of sub-processes present in the control process. The contribution of the new data function types, i.e. the classification of the single occurrence group of data is not experienced in this case, therefore unknown.

The FFP technique appears to be useful if applied at a higher level of abstraction (represents the functionality delivered to the system clients). This type of sizing ignores any internal triggers or processes that occur in a system due to the argument that these internal processes are more likely to be caused by system design rather than end-user functionality. This is useful for comparing projects implemented in different environment because of technical independence. However, there is limited usefulness of the above view of an organization internal evaluation purposes since more often than not, the development of the system is assigned at the component level.

5.4 The practical aspect

It is generally agreed by researchers the application of function point techniques does require expertise and experience. Function point counting requires many decisions to be made to resolve the ambiguity that arises during the counting.

It was the most difficult phase of all because of the need to study how the system operates. This implies and suggests that an organization use their internal personnel to conduct the measurement because not only do these people understand how the system works, they also have a better understanding of how the measurement should be conducted to be useful for the organization. Once the context of the measurement has been defined and the basic concept is understood, the identification and the assignment of function points to the data and transaction function types is relatively easy.

Good measurement practice is essential in today's IT development, it was discovered that implementing a metric program such as this case could offers advantages in detecting any inconsistency in the documentation. In measurement, it is used throughout the software development life cycle, not only better management or control of projects can be achieved through more accurate measure of effort, but also improvement in documentation quality.

From the literature, one of the noticeable improvements in this field over the past 5 years seems to be in the increasing acknowledgement of the complex nature of software size estimation and that the current improvement in the field is increasingly relevant to the industry practice. The introduction of structure in separating software layers and recognition of the impact of the different viewpoints to function point counts (e.g. business enterprise viewpoint, project view point) by Mark II counting manual (UKSMA 1998) is a typical example of increasing relevance to the industry practice. The current release of COSMIC-FFP (v2.2) recognizes the difficulties in setting up a software model to be used during the measurement process. It attempts to improve this aspect by introducing new steps to be performed to establish measurement context prior to the counting exercise.

5.5 Limitations

The obvious limitation in this study is the focus of one small enhancement project; this limits the external validity of the findings. The issues of boundary cannot be thoroughly explored in the case study. The impact of boundary definition and level of abstraction cannot be seen in the study because the enhancement project selected did not result in any changes to the data flow between internal processes. Only the application layer is affected in this project, thus sizing of functionality delivered by the different software layers is also not experienced in this case.

This study took a simplistic view of the system being studied. The assumptions might seem "superficial" in certain areas such as the data structure used for the counting purpose. The study assumed that the data used in the enhancement project was newly developed because the "true" data structure that the organization used was too complex. Therefore this has an impact on the counts not being fully accurate in representing the system. However, for the purpose of studying issues of boundary definition and level of abstraction used had no impact when this assumption is applied consistently throughout the counts.

This study took two different perspectives at sizing the system, i.e. at the top-level system view as well as the component level of system view. Nevertheless, it might be still possible to have different measures for the system other than from those views which again dependent upon the describer.

5.6 Further Research

There is the argument that related to user functionality can be further broken down into functional requirements and non-functional requirements which need to be taken into account in order to better explain the effort in developing the software product. Currently, the non-functional requirement is represented by the general system characteristics (GSC) and value adjustment factor (VAF) in both FPA and COSMIC FFP techniques. However, many researchers have found many of the GSCs are not relevant to the current technology. It is therefore further research for new system characteristic that relates to real-time system might be appropriate. [13]

There is also a need for further improvement in the counting manuals to incorporate generic practical application of the technique. It is very important to better guide both practitioners as well as researchers in applying the technique to practical situations and also be able to establish the appropriate context for measurement purposes.

6 Conclusions

This study has presented the applicability of full function point method for estimating software size in the domain of real-time software. By applying the FFP method to a practical application, in which the issues of counter subjectivity in establishing the counting boundary and the different system views for measurement purpose were explored. The study showed that a boundary played a very important role in the assignment of function points to a system. This is because the technique is highly dependent upon the concept of boundary in identifying both data and transactional function point types. The results from this study have demonstrated that a system could have different types of interaction with its users (human or devices) or other software. This suggests that different people can perceive the software functionality differently and that software sizing adapting only one perspective can be misleading depending on its use. One type of system size alone cannot describe the system's functionality completely. Different sizing is necessary, depending upon the different purposes or needs.

It is interesting to note that the function point counts derived from different models showed the same results. As can be seen from the total function points generated from this study (21 FPs), the enhancement project involved was very small in size. Thus it is unlikely to see noticeable changes in the various system components. Also the changes introduced by implementation of this project did not result in change to the interactions (boundary 1 and 2), therefore cannot be observed. It could be concluded, as long as there is no change to the interaction between the components, the result of both the different boundary definition and the level of abstraction used will be the same. This implies that top-level sizing would be adequate to determine the overall total functionality that a project delivers. However, software development budget for each components are often allocated to different teams, in this case we might consider using the lower-level sizing to gain specific estimate for each components.

It is also found that worldwide experience in this field is limited, requires the cooperation of both researchers and practitioners in the fields to share experiences to stimulate further development as

to achieve an “acceptable” level of the acceptance from both a practical and theoretical perspective. Practitioners and researchers must be able to understand the current limitation of the models so that they know how to use them safely and recognize that the application of the function point model may be consistent and useful provided the resulting measures are used in a relative context.

7 Acknowledgements

The authors would like to thank this telecommunication company for providing financial support and contribution to the context for this study and their staff for participating in the study.

Funding for this project was also provided by National Information and Communication Technology Australia (NICTA) and the University of New South Wales.

8 References

[1] A. Abran and P.N. Robillard, Software Management Based on Software Deliverables, *CIPS/CATA Congress 90*, 1990, 237 –245.

[2] A. Abran, J-M. Derharnais and S. Oligny, Measuring the functional size of real-time software, *Universite du Quebec a Montreal and Software Engineering Management Research Laboratory C.I.M Presentation*, CA, 1999, 1-104.

[3] J-M. Desharnais, D. St-Pierre, S. Oligny and A. Abran, Software Layers and Measurements, *International Workshop on Software Measurement(IWSW)*, La Superieur, Quebec, 1999, 1-8.

[4] L. Garcia and F. Quek, Qualitative Researcher in Information Systems: Time to be Subjective?, *Proceedings of the IFIP TC8 WG 8.2 International Conference on Information Systems and Qualitative Research*, Philadelphia, USA, 1997

[5] T. Hasting, Adapting Function Points to contemporary software systems: A review of proposals, *Proceedings of ACOSM' 95*, 1995, 103

[6] International Standard ISO/IEC 19761:2003, "COSMIC-FFP Measurement Manual version 2.2 - The COSMIC Implementation Guide), January 2003

[7] D.R Jeffery, and J. Stathis, Function Point Sizing: Structure, Validity and Applicability, *Empirical Software Engineering Vol.1*, 1996, 11-30

[8] D.R Jeffery, G.C Low, Function Points and their Use, *the Australian Computer Journal*, Vol.29, NO.40, Nov, 1997, 148-156

[9] C. Jones, Sizing up Software, *Scientific American*, December 1997, 1-7

[10] B. Kitchenham and K. Kansala, Problems with Function Points, *National Computing Centre and Laboratory for Information Processing*, 1996, 1-6

[11] P.D. Leedy, *Practical Research: Planning and Design*, 6th Edition, Prentice-Hall, New Jersey. 1997

[12] C.J Lokan, *An Empirical Analysis of Function Point Adjustment Factors,* School of Computer Science, Australian Defense Force Academy, University of New South Wales, December 1998, 1-15.

[13] C.J. Lokan and A. Abran, Multiple viewpoints in Functional Size Measurement, *International Workshop on Software Measurement (IWSM'99),* Lac Superieur, Canada, Sept 8-10, 1999, 121-132.

[14] Mark II Function Point Analysis, Counting Practice Manual v.1.3.1 UK Software Metrics Association (UKSMA), Sep 1998.

[15] C. Marshall and G.B. Rossman, *Designing Qualitative Research, 2nd Edition*, Sage Publication, California, 1995.

[16] M. Maya, A. Abran and P. Bourque, Measuring the Size of Small Functional Enhancements to Software, *6th International Workshop on Software Metrics*, University of Regensburg, Germany, Sept 19-20, 1996, 1-8.

[17] M. Maya, A. Abran, S. Oligny, D. St-Pierre and J-M Desharnais, Measuring the functional size of real-time software, *ESCOM-ENCRESS-98*, Rome, 1998, 191-199

[18] P. Morris and J.M. Desharnais, Measuring ALL the Software not just what the Business Uses, *IFPUG Fall Conference*, Orlando, Florida, Sep 21-25, 1998, 1-17.

[19] S. Oligny and A. Abran, Field Testing Full Function Points: Recent Results, *NESMA Autumn Congress*'98, Nov, 1998, 1-24.

[20] G. Rule, System Size, Component Size and Project Size; Appropriate Measures for Distinct Purposes, *European Software Cost Modeling Conference (ESCOM 1994)*, Ivrea, May 11-13, 1994, 1-17.

[21] R.E. Stake, *"Case Studies", Strategies of qualitative inquiry, Handbook of qualitative researcher.* 2nd Ed., Thousand Oaks, Sage Publications, California 1998.

[22] C.R. Symons, Come Back Function Point Analysis – All is forgiven!, *Conference on Software Measurement and ICT Control (FESMA-DASMA2001),* Heidelberg, Germany, May 9-11, 2001.

[23] C.R. Symons and P.G. Rule, Once size fits all - 'COSMIC' -Aims, Design Principles and Progress, *Proceedings of the 10th European Software Control and Metric Conference (ESCOM SCOPE 99),* Herstmonceux Castle, England. 1999, 1-11

[24] C.R. Symons, Function Point Analysis: Difficulties and Improvements, *IEEE Transaction of Software Engineering,* Vol. 14 No.1, Jan, 1988, 2-11.

[25] S. Treble and N. Douglas, *Sizing and Estimating Software in Practice: Making MKII Function Points Works*, the McGraw-Hill International Series in Software Engineering, London. 1995