Forgotten Aspects of Function Point Analysis

Authors: Paul Goodman & Pam Morris

Email: paulgoodman@sms-tbl.freeserve.co.uk
Email: Pam.Morris@totalmetrics.com

This article contains material from both the International Function Point User Group Counting Practices Manual (version 4) and the UK Function Point User Group Mark II Counting Practices Manual. This material is reproduced with the permission of those organisations.

The title of this paper stemmed from a presentation at the International Function Point User Group (IFPUG) conference in the spring of '94. A strong theme of that presentation, given by one of the authors of this article, Pam Morris, was simply that there is a very important word in the term "Function Point Analysis", namely "Analysis". Subsequent discussions between the authors and others have led to this article which we both hope will trigger some thought and response.

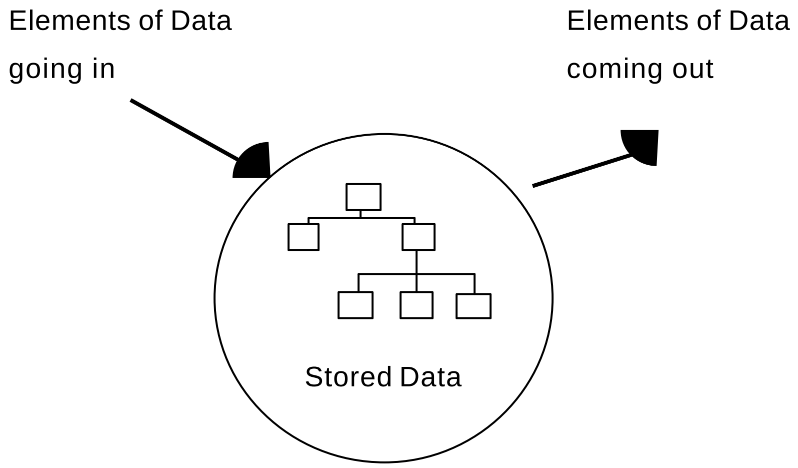
Generally, we tend to think of FPA as providing us with a size measure and the Function Point Index and when talking about the uses and benefits of FPA, we talk about the index and the size measure as being something that enables meaningful comparison of, for example, productivity across projects, systems and enterprises. Because of the observed relationship between size, in terms of the Function Point Index and effort in many organisations, we also concentrate much of our attention quite rightly on the use of FPA in cost estimation. These are the traditional uses of FPA.

Bearing in mind the term "Analysis", we wish to look at some of the fundamental principles of FPA and see if there are other uses.

When you think about FPA, what is the most important factor that differentiates this from other techniques in software engineering? It has to be that FPA views the system from the users' perspective; the result of applying FPA is the derivation of a measure of the "functional user requirements" (ISO Draft International Standard). Remember that point, FPA is based on the users' view rather than the developers' view.

Now how is this measure derived? We are going to talk in terms of new systems but much of what we will say applies equally to enhancement projects. The first thing we do is to take a view of the system as shown in figure 1.

figure 1



Essentially, this is how customers see computer systems. They put information in and they get information out and they recognise discrete groups of data, entities or files that the system will use and maintain. In fact, this process does not even need to be supported by a computer system. Think of the Dickensian picture of a clerk working in a counting house at the big, old, high desk with wooden pigeonholes to store data. **That business process can equally be represented by the model in figure 1**. Hold that thought also and we will return to it later.

The next stage in FPA is to model the users' functional requirements. In other words, we will uniquely express the general model given in figure 1. The way we do this is to break the process down through a number of layers. A system supports a business function, maybe accounting and this can be divided into a number of business areas. We could define a business area as a users' recognisable segment of a business function that, for business reasons, may be supported separately. An example may be end of year accounting.

If we keep decomposing our system, as we must if we are going to apply FPA, we reach something I will call the business operation level. A business operation can be thought of as a set of one or more activities which, when taken together, form a complete, user recognisable business procedure. In accounting, an example could be to complete the end of year tax return which is part of end of year accounting. Now, where is all of this leading us?

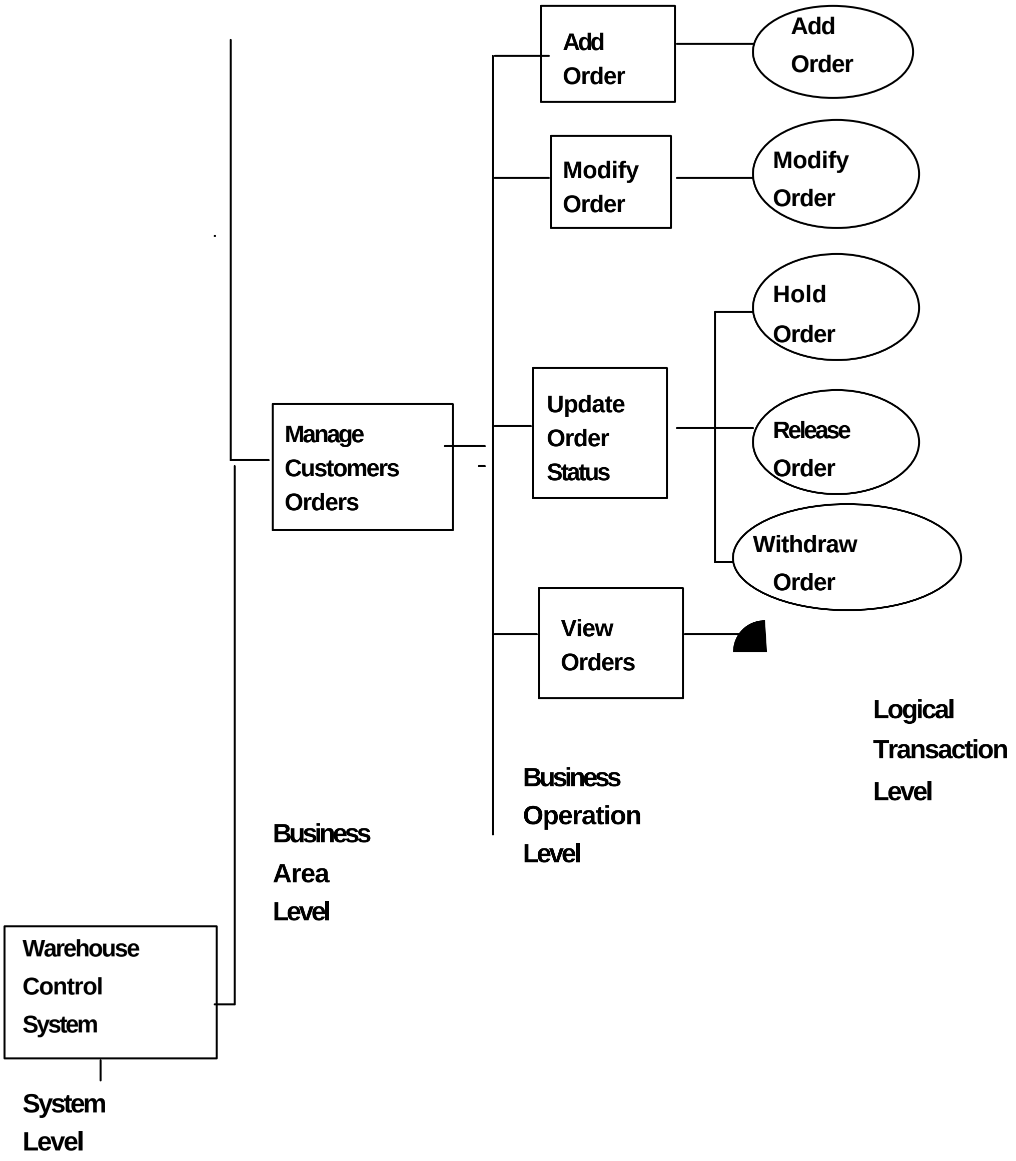
So far, we have taken a view of the business function that reflects the way people think about what is going on. Now we move slightly away from that view to more of a systems oriented view. Decomposing still further we reach the lowest level of user recognisable functionality. That is, "...the lowest level business processes supported by the system..." (UFPUG CPM).

And before going any further we have to digress. Some of you, seeing the source of that quotation will think that what we have said so far may relate only to Mark II FPA. Let me give you another quote and note the source, "...the smallest unit of activity that is meaningful to the end user of the business..." (IFPUG CPM). The IFPUG CPM calls these "elementary processes", UFPUG call them "Logical Transactions". There are some differences in definition **but the spirit is the same.**

We will stick with the term Logical Transaction for no better reason than the fact that the editor of this piece had it in the next diagram already and is a strong believer in re-use!

Having got down to the level of the Logical Transaction we have a functional decomposition of users requirements that define a process that may be supported by a computer system. Figure 2 shows this for part of a warehouse system.

Figure 2



Once we reach the point of sizing Logical Transactions by identifying information that enters and leaves the process and the data stores acted on, whether you use the IFPUG or the Mark II approach, you have not just a means of deriving the Function Point Index, **you have a pure expression of the user's functional requirements**. Think about that for a moment. In order to apply FPA you have, for example, defined not the report layouts but the information that will be contained in an output which may be a report. That is all you need for FPA. In fact, one of the problems FP analysts have is that so called requirements specifications often contain lots of unnecessary information; express how the system will do things rather than what it will do and are incomplete.

This use of FPA to define concise yet rigorous specifications of requirements can lead to a fuller understanding of what is going to be undertaken much earlier than we in the IT industry are used to. The benefits of this are immense, as we all know the increased cost of fixing things later. Also, with the increasing use of third party software development and support, well-defined specifications that can be sized naturally become the basis for contracts. Pity the poor project manager who has been signed up to a fixed price contract only to find that half the requirements have been misinterpreted or even missed altogether!

There is a further ramification to the analysis aspect of FPA. Every time you perform a Function Point Analysis, you are engineering, often reverse engineering, a form of requirements specification from available information. You are, in effect, duplicating the requirements specification process. Given that the specification process is still regularly cited as the biggest problem in software engineering today raises a question.

Can Function Point **Analysis** be used as the solution to the problem of requirements specification? From experience, we know that the answer is a partial yes and that the part of the problem it addresses is the most significant, that of deriving the functional requirements. Identifying constraints etc. can easily be added to the process of FPA.

There is then the fact, we suspect, that FPA can be used to size any business function whether or not that function is supported by a computer system. That means it could be used to size a business in terms of functional requirements. To be able to do this implies that a model of the business needs to be constructed that is amenable to the application of FPA and this can be done in the same way that a model of the users logical requirements is derived in order to apply FPA to an IT based system. This business model can supply valuable information about the business. After all, how many senior managers have any idea what the totality of their business function is? If it was done then a logical next step would be to do the same thing to the systems that support the business function. In other words, from the physical systems that exist, reverse engineer the functional requirements (we've deliberately dropped the word user here) that model what the system(s) do. Then compare the two.

Pie in the sky? No, this has already been done in at least one Australian organisation of some size and what did they find? Whole areas of user functional requirements, i.e. business functionality, that was unsupported. Massive duplication of supported functionality, implying significant maintenance costs that were unnecessary. Might it be worth thinking about applying FPA to your business in this way? Saving millions is a powerful incentive!

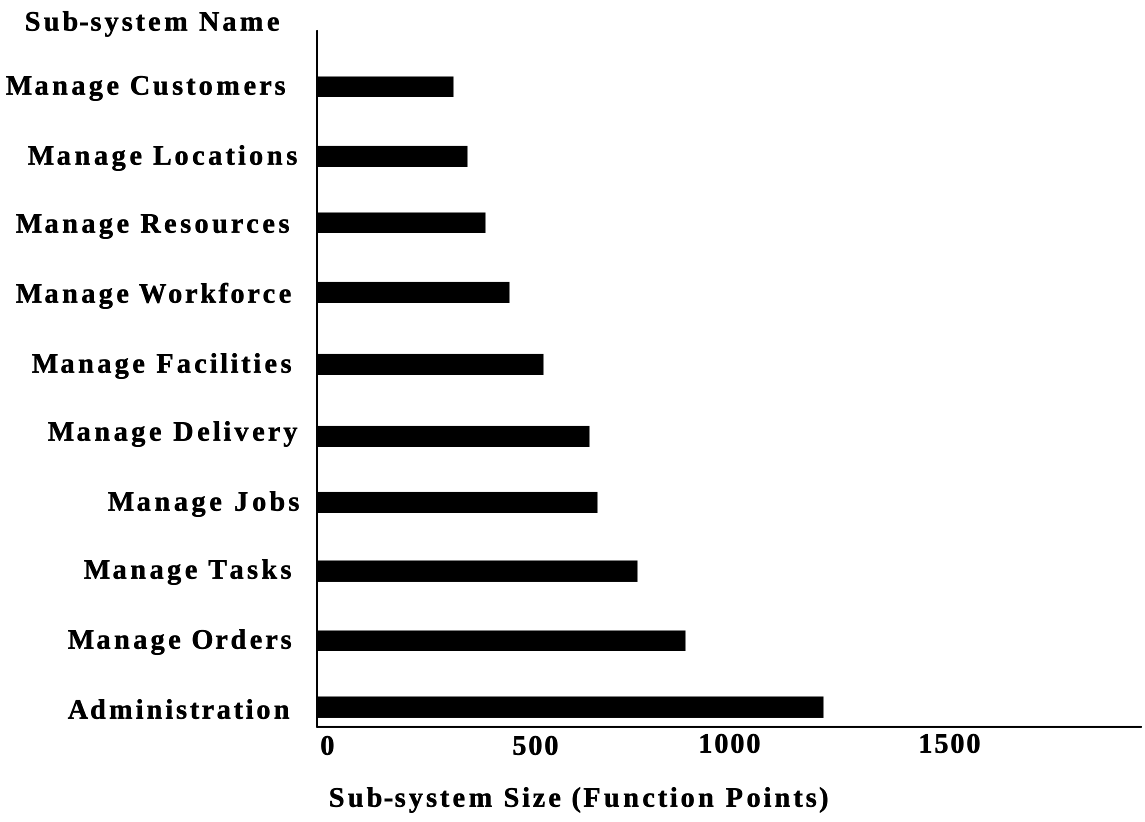
So far, we have looked at two "forgotten" aspects; the model FPA provides for functional user requirements and the fact that it can be applied to business processes as well as the systems that support those business processes. Let us now consider a third aspect. Taking one step back from the Function Point Index we have the Logical Transactions, or elementary processes. These provide us with a different view of the job in hand, the job of building the system to support the functionality requested. Perhaps there is something in this that could be used to help the project managers.

Many of us have been asked to report progress in terms of "percent complete". This is sometimes expressed in terms of components built vs. components to be built; actual lines of code delivered vs. estimated lines of code to be delivered; actual effort consumed vs. estimated effort still to be consumed (which seldom decreases!) or various combinations of these. These measures have serious defects. They tend to be meaningless until quite late in the projects life, they are often based on estimated values and, if project managers are honest and we have both been members of this much maligned breed so we again speak from experience, the reports are often best guessed.

Yet with the model we have in FPA, at the Logical Transaction level, is practically ideal for progress monitoring. Once we have the model, expressions such as 50% of our Logical Transactions have now been signed off review in technical design, or even 25% of the total of our total Function Point Index have now been signed off review in technical design are meaningful. The model, the thing that has enabled us to do my FPA, is not subjective (despite what some people claim, this has been proved at least to the satisfaction of most, by more than one study); is available early in the life cycle and is not an estimate. It is true that the Function Point Index may change but only if the requirements change. The old chestnut that "the user doesn't know what they want until we have built or at least designed the system" is a cop out that should be rephrased as "we (IT) are lousy at talking to our customers to find out what they want".

We do not even have to stop with progress monitoring. The Function Point model can help us manage our projects better. Each Logical Transaction has an FP index of it’s own. This can be seen as a measure of how difficult it is going to be to build, not for small variations in size but certainly for larger variations. We can even assess whether it is entity heavy or input or output heavy. Given this information it may make more sense to give specific transactions to specific people with specific skills. Of course, when we start to group transactions to the business operation level, that is clusters of Logical Transactions or elementary processes, this management approach can make even more sense.

Figure 3



Where would you put your more experienced staff?

We firmly believe that we have only started to scratch the surface as far as the way in which we use FPA is concerned. Perhaps what we have called the "forgotten aspects" are really "undiscovered uses". As with many things, if we look at the fundamental principles of FPA and try to concentrate less on what we currently use it for, those principles may lead us to those undiscovered uses.

If you are currently using FPA in unorthodox ways or would like to talk these ideas through in more detail with a view to implementing them, the authors can be contacted by email, telephone or fax.

Tel : +61 (0)3 9882 7611
Fax : +61 (0)3 9882 7633
Email : admin@totalmetrics.com