uTip

Guidance from the
Functional Sizing
Standards Committee

uTip - Early Function Point Analysis and Consistent Cost Estimating

uTip # 03 – (version # 1.0 *2015/07/01)*

Author: Adri Timp		
Reviewers:		
Diana Baklizky	Daniel French	Roopali Thapar
Bonnie S. Brown	Steve Keim	Peter Thomas
E. Jay Fischer	Tammy Preuss	Charles Wesolowski

uTips (Usage Tips) provide insight into potential uses of function points to support an organization's business needs. While uTips provide insight on usage opportunities, they do not provide detailed direction on the application of the IFPUG FPA method in a particular situation. When necessary, the uTip maybe be followed by additional content on the topic providing specific how-to guidance. uTips are not rules, but rather interpretation and application of the rules, and provide guidance using a realistic example to explain the topic being covered.

This uTip is focused on describing the IFPUG FPA method as it applies to sizing and cost estimating in the early stages of the software development lifecycle, as well as sizing in a fast way. This uTip is not an exhaustive examination of the subject. At the end of the paper, suggestions for further reading are provided.

Introduction

Myths

- "You cannot apply FPA in early stages of the software development process, so in the practice of budgeting software development FPA is useless."
- "You need a high level of detail of the functional user requirements before you can successfully apply FPA."
- "Cost estimating and budgeting using FPA takes lots of time. It's not worth the effort".

No! No! No! These three widespread misunderstandings prevent people from benefiting from FPA at virtually any moment!

The CPM [1] states, that FPA estimates are possible by making assumptions about unknown functions and/or their complexity in order to determine an approximate functional size (part 2, page 3-8).

This uTip provides an introduction on how to apply FPA in early stages of development or enhancement projects, as well as how to perform FPA very quickly using estimating techniques. It also shows how to maintain a consistent size and cost estimation approach throughout the software development lifecycle, taking into account autonomous growth and scope creep.

Estimating the functional size

The two major reasons to estimate the functional size of a development or enhancement project instead of performing a detailed FPA are:

- because the details of the functional user requirements are not known
- to significantly speed up the FPA-process

Two main estimating approaches are the High Level FPA Method and the Indicative FPA Method.

High Level FPA Method

If the functions the user wants are identified, but the details of the logical processing and the logical files and DETs involved are not known, the advice is to look from a bird's-eye view at the functionality and perform a high level function point estimate:

- determine all functions (ILF, EIF, EI, EO, EQ)
- rate the complexity of an ILF and EIF as Low and an EI, EO, EQ as Average
- assign the function points and accumulate

The only difference from the detailed function point estimate is that complexity is assigned by default.

Indicative FPA Method

While the functions a user wants must be known for a high level function point estimate, sometimes very little is known about the application besides what data will be maintained. There are also situations where there is a need for a very rough estimate of the size very, very quickly. Using the indicative functional size approach provides a rough estimate of the functional size based on the likely logical files (e.g., Customer, Invoice, Payment):

- determine all data functions (ILF, EIF)
- the indicative functional size = 35 x number of ILFs + 15 x number of EIFs

This calculation is based on a projected ratio including likely transactions for each data function and experience has shown that it is a suitable approximation. The result may be considered a ROM (Rough Order of Magnitude).

When to use which method?

A detailed function point estimate provides a more accurate functional size than a high level or indicative function point estimate, but it also requires more time and needs more detailed specifications in order to determine RETs, DETs, and FTRs. The phase in the software development life cycle, the business use for the data, timing requirements, availability of information about the application or project, the type of project, and availability of personnel – both subject matter experts and function point analysts – may influence the decision as to which sizing technique is appropriate.

Where the accuracy of the size is an absolute necessity, a detailed function point estimate provides the detail. If performing a detailed function point estimate early in the lifecycle, it is important to recognize that the RETs, DETs, and FTRs may not be correctly identified, resulting in a functional size that is skewed higher or lower.

When information is not available to accurately determine RETs, DETs, and FTRs, it is necessary to estimate the complexity of application functions. There are also situations where a business decision is made that the level of accuracy of a detailed function point estimate is not required to meet the business purpose of the functional size. In these cases, the high-level sizing approach can be used.

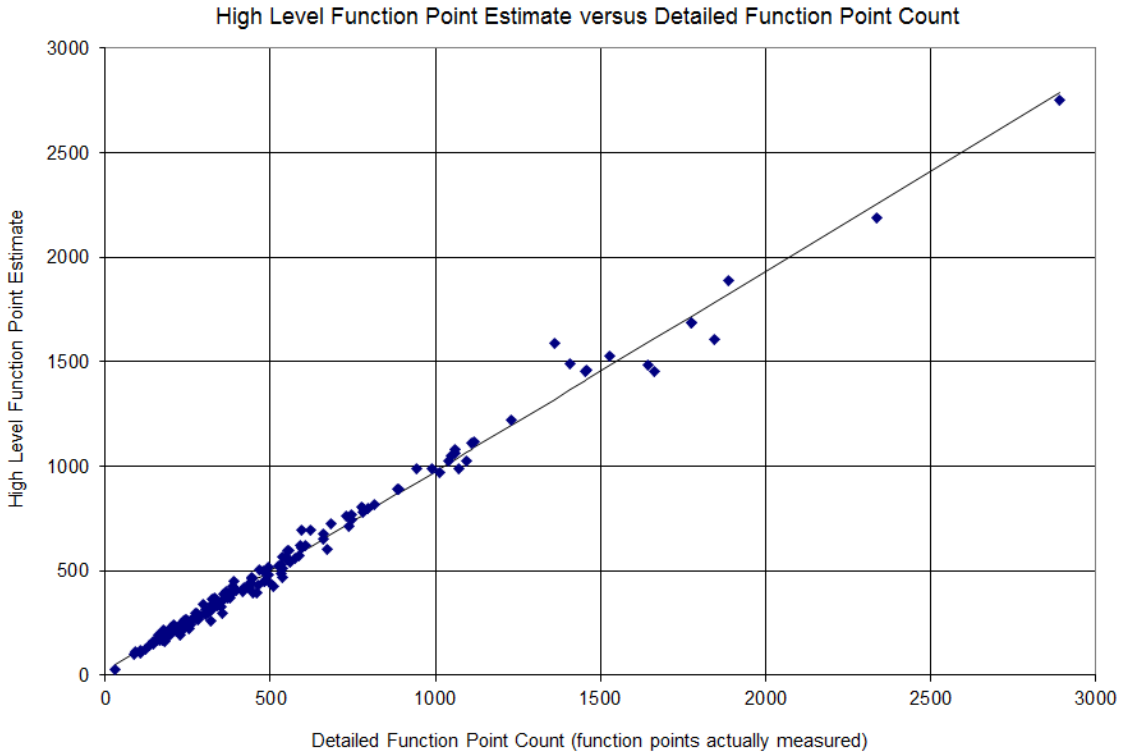
For some business purposes an indicative function point estimate provides a reasonable estimate of size. It is relatively easy to perform an indicative function point estimate because a high level data model is often available or can be created with little effort. However, the indicative method is not appropriate for estimating enhancement projects.

During the software development life cycle, whatever method is used, the result is always an approximation of the functionality to be delivered. The CPM [1] states, that it is essential to update the functional size upon completion of the project (part 2, page 4-4).

How much worse is it?

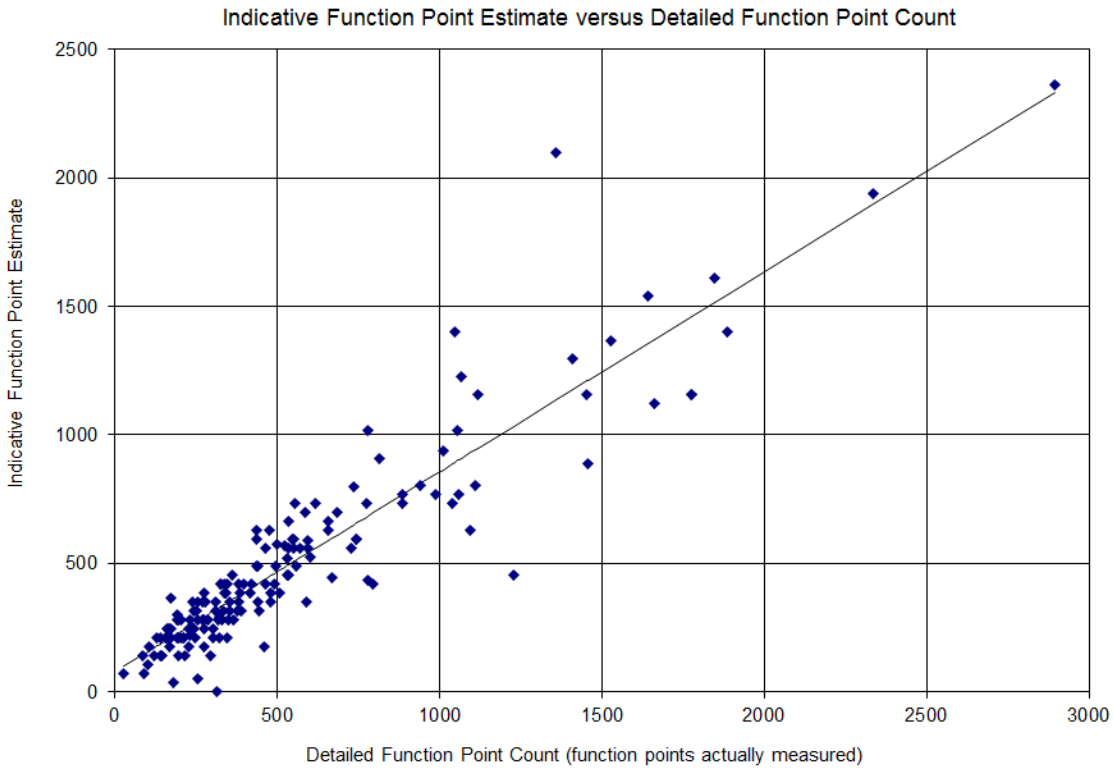
One might think "These estimates cannot work out!" In reality, these approaches are not that bad, and even quite good!

Using a database of about 100 developed and implemented applications research has been done by Nesma on the accuracy of the high level and indicative function point estimates [2]. The implemented applications were simultaneously measured using all three methods. The results are presented by Nesma in two graphs. We see a good correlation (straight line) in both cases.



This graph shows that the outcomes of a high level FPA and a detailed FPA are typically very close. Many companies use high level FPA only because they believe there is no statistically significant difference in the results of both approaches. If it works for a company, it works. It significantly speeds up the sizing process and might lower the threshold within an organization to accept FPA. But even more importantly, it enables you to size in early stages of development!

The same research showed, that also the results of the indicative estimate are quite good; however, there may be considerable deviations (up to about 50%) in some cases. That is why one should be careful using the indicative function point estimate. The strength of the indicative method is that one easily gets a rough estimate (the rough order of magnitude) of the size in a very short timeframe.



Autonomous Growth and Scope Creep

Growth is the increase in function points later in the project. It may occur due to autonomous growth and due to scope creep.

Autonomous growth [3] occurs through revealing functionality while detailing and having a closer look at the functional user requirements; it concerns functionality that was already included in the requirements, but was not originally recognized.

Scope creep occurs through the addition of new requirements by the user. It generates function points that would not have been found even after detailing the requirements. An example of scope creep is "After due consideration, I also want to have the option to delete customers."

There are typically stable percentages of autonomous growth within organizations where software is repeatedly developed in the same manner. The percentages can be determined by recording the functional size at the various phases of development that can be discerned within the organization. If the percentages are known, they can be used for a consistent size and cost estimation approach.

Growth in function points due to autonomous growth is not attributable to additional user requirements, but just a change in counted function points later in the project. It is a (virtual) growth in function points, but not a growth in user requirements.

Growth due to scope creep is easier to manage and to understand. If a user adds user requirements, it is a change, the size of the product really changes, function points are added, and the impact to the budget should be addressed.

Accounting for Autonomous Growth and Scope Creep in Sizing and Cost Estimating

For an effective use of FPA it is important to maintain a consistent size and cost estimation approach throughout the software development lifecycle. For consistency and credibility, take into account the effect of autonomous growth (revealed function points) throughout the software development lifecycle and the potential impact of scope creep.

For effective and consistent size and cost estimation organizations should track growth percentages from one phase to the next.

The following example illustrates autonomous growth for a fictional organization with a traditional non agile software development life cycle:

Phase	Estimated functional size in this phase	Empirical percentage autonomous growth after this phase	Projected functional size of the product	
Feasibility Study	100	40%	140	Please note: these % values are **for illustration purposes only**.
Analysis	117	20%	140	
Functional design	127	10%	140	
Construction	140	0%	140	
Test	140	0%	140	

Function point analysis is performed during the Feasibility Study and the estimate is 100 function points. Based on this organization's historical data, further refinement of the requirements should reveal 40% extra function points that were already included in the requirements, but could not be "seen" in that early stage of the project. The projected functional size of the product is 140 FPs.

Effort estimates should be based on the projected functional size of the product. If an organization has a productivity rate of 10 hours / FP and the FPA during the Feasibility Phase would result in 100 FPs, the effort estimate would be (100 + 40%) x 10 hours / FP = 1400 hours. Accounting for autonomous growth reduces budget

overruns due to the growing number of function points even though the application and the user requirements do not grow.

It is also common that size grows due to additional requirements by the user (i.e., scope creep). This is real growth which is different than autonomous growth. To address scope creep, the project can set aside an extra budget in function points for the user to specify eventual extra functionality (new function points) later in the project. Planning the project taking into account the scope creep reduces the schedule and budget impact that will occur when the user requests additional functionality. The functionality should be managed using an effective change management process that estimates impact to size, cost and schedule for the additional scope.

Summary

The High Level FPA method can be used to size an application early in the software development life cycle. Taking into account the effect of autonomous growth of each phase one gets a fairly stable function point size throughout the project.

The High Level FPA method can also be applied as alternative to a detailed FPA estimate: the outcome is not significantly different, while the time to execute the FPA is considerably less, thus increasing the acceptance of IFPUG FPA as method for sizing and cost estimating.

The indicative FPA method may be used to get a very fast, rough indication of the size of a project or an application. It is not suited for contractual commitments.

Method	**Accuracy**	**Cost / time to perform**	**Identify all data functions and transactions**	**Use complexity rules from CPM**	**Autonomous Growth**	**Scope Creep (see [1], part 2, page 4-4)**
Detailed FPA (full CPM)	High	High	Yes (no assumptions)	Yes	Needs accounting for – e.g., add 40%, depending on phase	Eventually set aside extra budget for extra functionality
High Level FPA	High	Medium	Yes (with assumptions)	No	Needs accounting for – e.g., add 40%, depending on phase	Eventually set aside extra budget for extra functionality
Indicative FPA	Medium/ Low	Low	Data: yes Transactions: No	No	Needs accounting for – e.g., add 40%, depending on phase	Eventually set aside extra budget for extra functionality

Frequently Asked Questions (FAQ)

1. How do I perform a high-level FPA, if I do not know the type of some functions?

   Knowing the number of functions is most important. If the type of a function (EI, EO or EQ, ILF or EIF) is not known, just assign 5 function points for each unknown function type.

2. How do I perform a high-level FPA if I am not sure I have identified all functions?

   Autonomous growth accounts for discovery of additional function points later in the software development life cycle.

3. How do I perform a high-level FPA if the user cannot specify which reports he wants, but thinks that about ten reports will be needed?

   As said, the most important factor is the number of functions. Just use that number!

4. I want to do a high level FPA. For some of the functions I have enough information to determine the complexity. Should I do that?

   No. The high level FP estimate method works thanks to "the law of large numbers." If complexity is assessed for some of the functions, the average is skewed.

5. In a contractual relationship with a supplier, can one use high-level FP estimate?

   Yes. As said, there is no significant difference in the results compared to a detailed FPA. For effective contracting it is far more important, that there is agreement about the number of functions, than about the complexity of each individual function. Discussions about complexity are very tedious, while the impact is low. Be more efficient and effective by focusing on the things that really matter: the functions and scope. Before you close a contract, agree which will be used - detailed or high level FPA.

   Whenever contractual relationships exist, it is imperative that all parties understand the implications of any approach used in the contract. These estimation techniques do not conform to all of the rules in the ISO standard for IFPUG function point analysis [1]. If a contract specifies a different approach than ISO standard IFPUG function point analysis, the implications of any variation should be clearly understood by all parties.

6. In a contractual relationship with a supplier, should an indicative FPA be used?

   We advise not to use indicative FPA for contract performance, because there might be significant deviations between the outcome of an indicative and a detailed or high level FPA. However, indicative FPA may be utilized as an early indicator of functional size or rough order of magnitude.

7. The Functional User Requirements in the Feasibility Study are, by definition, high level. I did my very best and counted 200 function points. Due to experience in the past in this company it is known, that the Autonomous Growth after this phase is 40%. So I drew up a budget based on 280 function points.
   In the analysis phase the user said, that he will now "order" an additional 80 function points, because of the "reserve" I created. Is that correct?

   No. The 280 FP is the (expected) actual size of the User Requirements as specified in the Feasibility Study. There is no room to order more requirements without paying. Compare it with a picture taken from a coast line on earth by a satellite high above the earth surface: the coast line seems to be quite straight, and the size might look like 2000 miles. But, by having a closer look and measuring on earth the coast line appears to have an actual size of 2800 miles.

8. Can high level FPA be used instead of detailed FPA even when the detailed requirements are available or the project has been implemented?

   Yes.

9. How do I know, what the autonomous growth for my organization is?

   It is important to have the evidence of an internal repository for demonstrating the autonomous growth trend and magnitude.

10. Should I update the functional size upon completion of the project?

    Yes. In addition to refining the autonomous growth rate, the final delivered functional size is a component of many useful metrics. Please refer to [4] for additional guidance.

11. What are the benefits of a detailed FPA?

    A detailed function point analysis provides a more complete definition of functional software requirements that should facilitate development, testing and maintenance of the software resulting in a higher quality product.

Further Reading

1.	Function Point Counting Practices Manual, release 4.3.1	IFPUG Publication ISBN 978-0-9753783-4-2
2.	Early Function Point Analysis (http://nesma.org/downloads/early-function-point-analysis-english/)	NESMA publication
3.	The application of Function Point Analysis in the early phases of the application life cycle	NESMA publication ISBN: 978-90-76258-20-1
4.	The IFPUG Guide to IT and Software Measurement	IFPUG Publication ISBN 978-1-4398-6930-7

IFPUG offers uTips at no charge to the international function point community to stimulate the further promulgation and consistent application of the IFPUG FPA Method. IFPUG would appreciate if you or your organization would support IFPUG in its mission by becoming a member. For further information about becoming an IFPUG member, please visit www.ifpug.org or send an email to ifpug@ifpug.org. IFPUG thanks you for your support.

