uTip

Guidance from the
Functional Sizing
Standards Committee

# uTip - Early Function Point Analysis and Consistent Cost Estimating

uTip # 03 – (version # 2.0 *2025/0x/xx)*


| **Author:** Adri Timp, Marcello Sgamma | | |
|---|---|---|
| **Reviewers:** | | |
| Esteban Sanchez | Diana Marano | Diego Ferreira Da Rocha |
| Andrés Gutierrez Poveda | Cleber Ferrareze | Carlos Vasquez |
| Daniele Zottarel | Luigi Buglione | |


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 may 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 document presents three methods for approximating FPA: High Level FPA, Indicative FPA and Simple Function points. This uTip is not an exhaustive examination of the subject. At the end of the document, 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).

Moreover, in 2019 IFPUG acquired a lightweight functional measurement method, compliant with the ISO14143-1 standard, to increase the acceptance of functional sizing in all communities of software developers: the Simple Function Point method. An IFPUG Task Force, including members of the FSSC and NFSSC, spent two years analyzing and experimenting with the method, and completed its task producing Version 2.1 of the Simple Function Points Manual in October 2021[5].

This uTip provides an introduction on how to apply FPA or SFPs 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 challenges like autonomous growth and scope creep into account.

## Estimating the Functional Product Size

The two major reasons to estimate the functional product 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

Some estimating approaches are the High Level FPA Method and the Indicative FPA Method. Simple Function Point (SFP) can also be used for that same purpose, since fewer details are requested by this methodology.

## High Level FPA Method

If the functions the user wants are identified, but the complexity details (DETs, RETs and FTRs) 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 between the High Level FPA method and the detailed function point count is that the former assigns complexity by default [2].

## 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) [2].

## Simple Function Points (SFP)

The SFP method [5] leaves out primary intent of transactions and position of logical data files (with respect to the boundary), focusing only on identifying the Elementary Process (EP) and the Logical Files (LF). There is no complexity classification or weighting of the Base Functional Components (BFCs as defined CPM).

The size of an EP is 4.6 SFP, while the size of a LF is 7.0 SFP. Therefore, the functional product size expressed in SFP is based on the number of data files (#LF) and the number of transactions (#EP) for the software application being measured:

$$Size_{[SFP]} = 4.6 \times \#EP + 7 \times \#LF$$

## When to use which method?

A detailed function point measure by FPA is the most accurate functional size when compared to any estimation or approximation, but it requires more time as well as more detailed specifications 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.

When 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 the purpose of the count can be satisfied with an estimate of the count (full FPA is not needed). In these cases, the high-level sizing approach or SFP

sizing can be used. These two methods are effective also for estimating enhancement projects.

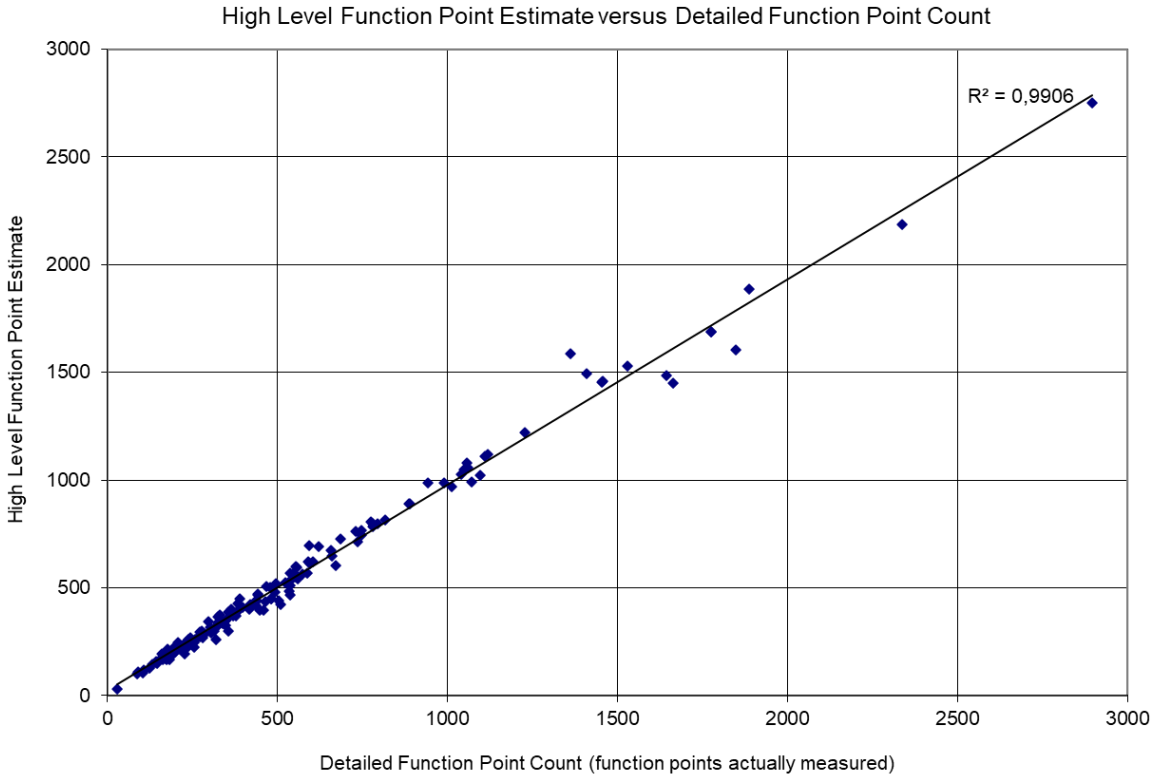
For some business purposes an indicative function point estimate provides a reasonable estimate of size. It is relatively easy to perform an indicative function point estimate because a high-level data model is often available or can be created with little effort. However, the indicative method is not 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 techniques won't work!" In reality, these approaches are not that bad, and even quite good!

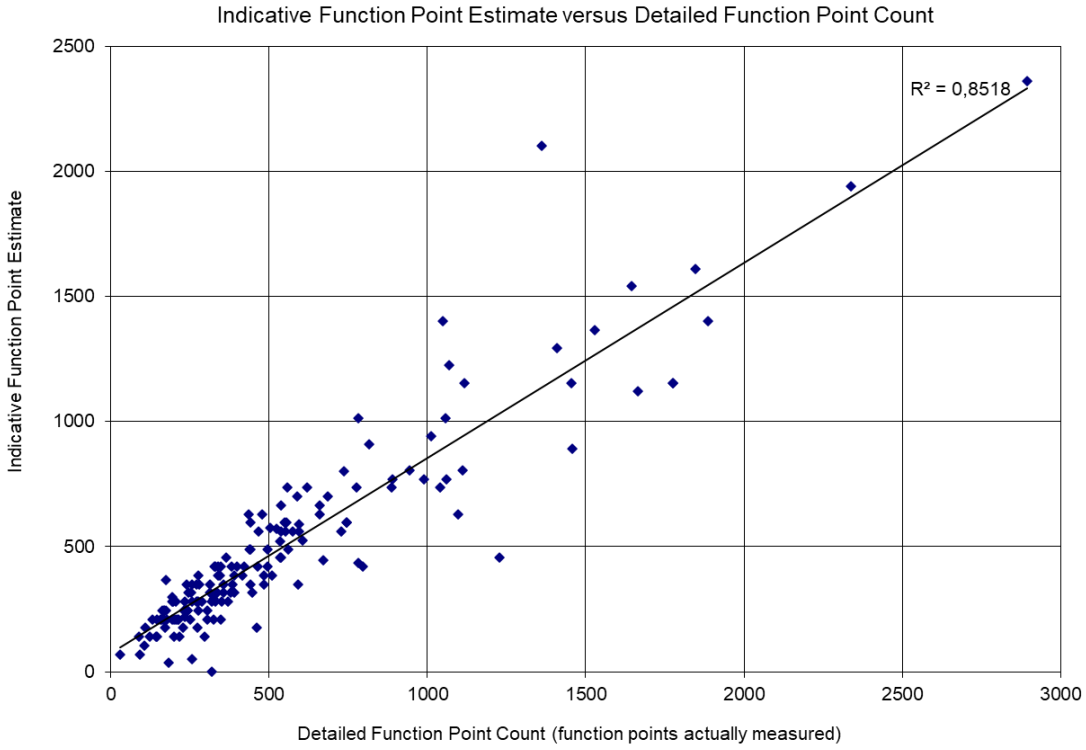
Using a database of about 160 developed and implemented applications, research has been done by Nesma on the accuracy of the high level and indicative function point estimates [2] when compared with detailed FPA. The implemented applications were simultaneously measured using all three methods. The results are presented by Nesma in two graphs. We see a good correlation ($R^2$ close to 1) in both cases.



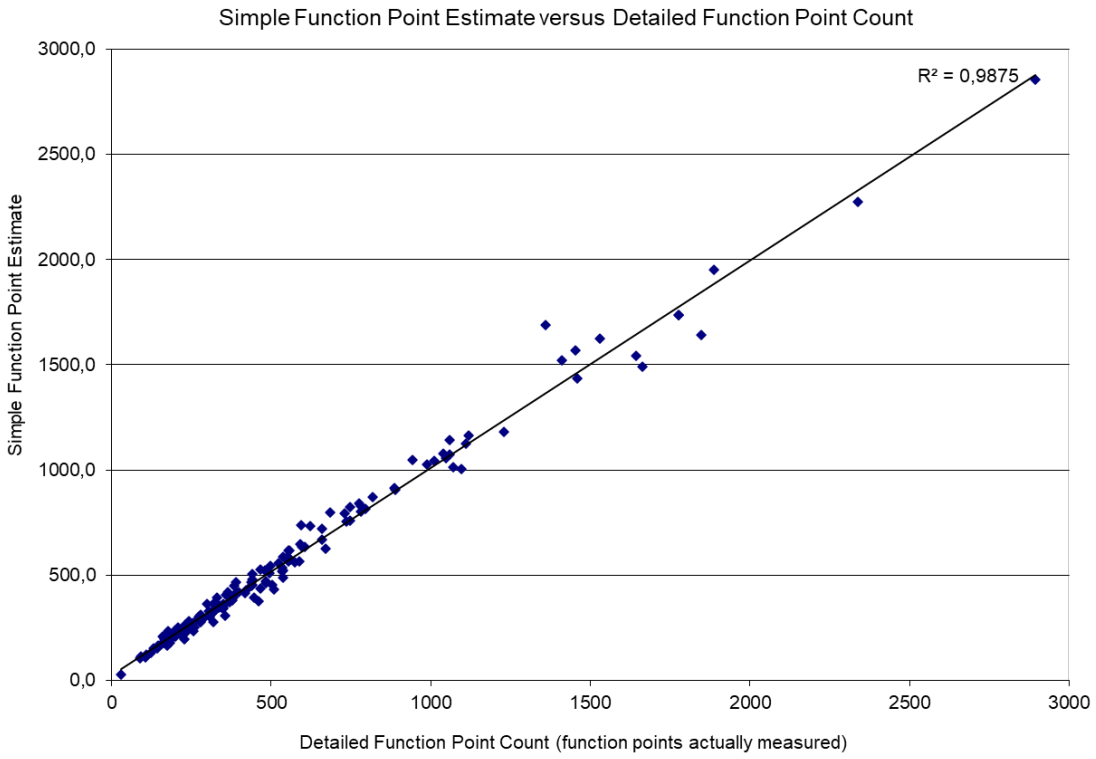

This graph shows that the outcomes of a high level FPA and a detailed FPA have a high correlation. Many companies use high level FPA to get a rough idea of functional size, as it significantly speeds up the sizing process and might lower the threshold within an organization to accept FPA. But even more importantly, it enables you to size in early stages of development!

The same research showed that also the results of the Indicative FPA Method 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

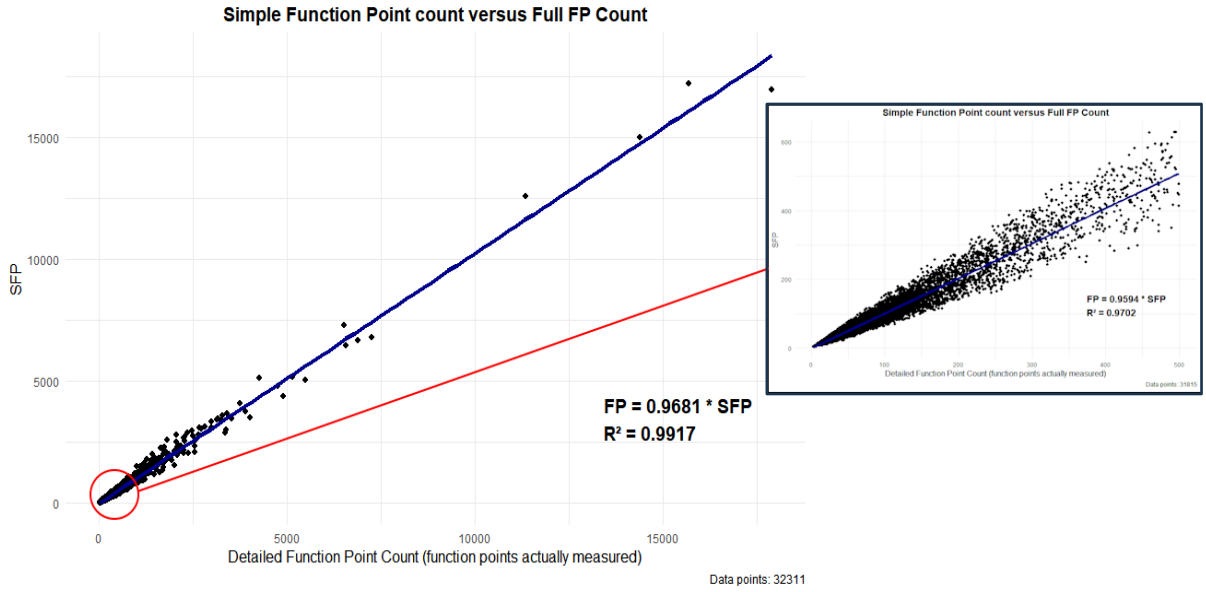
estimate (ROM - Rough Order of Magnitude) of the functional product size in a very short timeframe.




Based on the same data (BFC count) from the Nesma study, an SFP measure has been produced for the same projects, and the graph below shows a substantial correlation between high-level FPA and SFPs in estimating functional product size.




A similar analysis has been done to demonstrate the correlation between Simple Function Points (SFP) and Full Function Points (FPs). A well-recognized metrics organization in Brazil has collected anonymized data from public and private contracts. The dataset comprises 32,311 measurement cases from 2,147 software applications, totaling 2,027,894 Function Points across 221 public and private companies. The following chart illustrates the correlation between the two counting approaches:



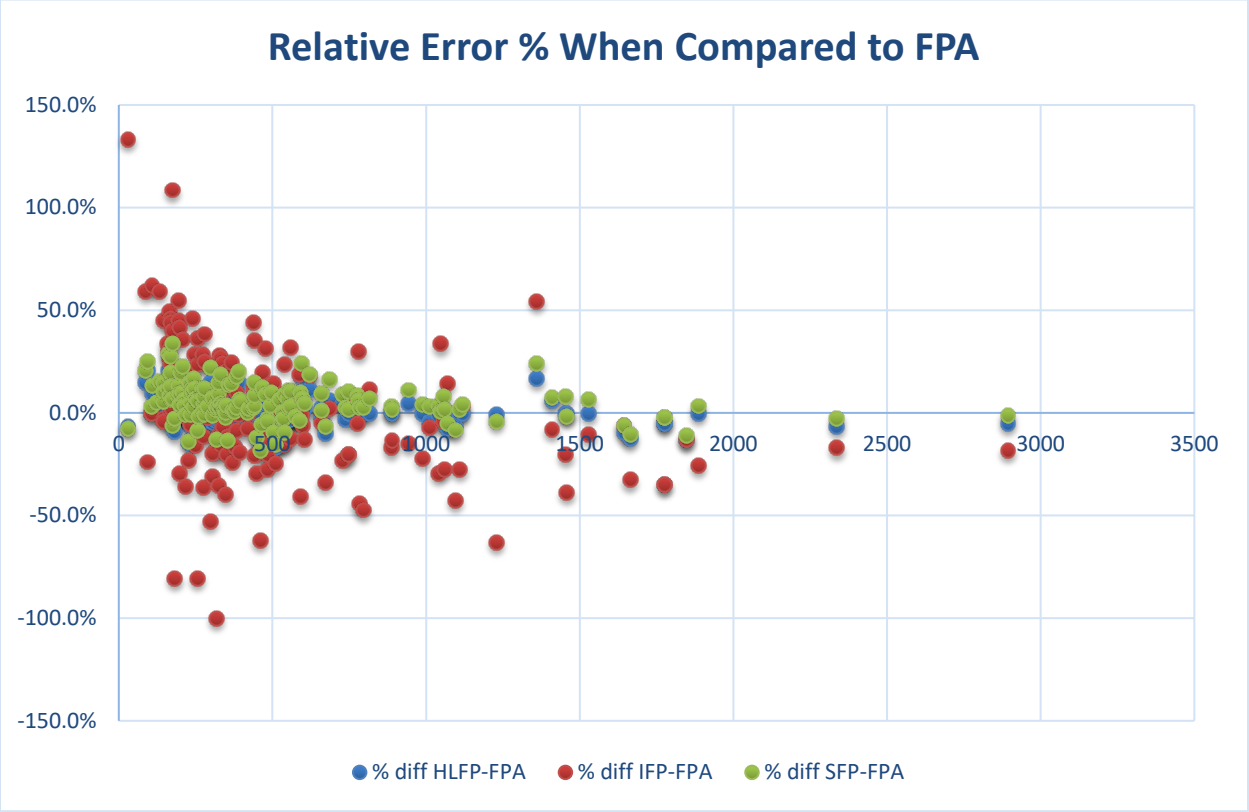
The chart above illustrates the high degree of correlation between detailed Function Points and Simple Function Points. The chart drills into the 0 to 500 FPs range for a more precise view. The SFP method can be applied early in the project when the requirements are not detailed enough to identify DETs, RETs and FTRs. In addition, the SFP count significantly accelerates the counting process.

## Caution When Using Estimation Methods

The three methods analyzed show good correlation with FPA, but the reader needs to be aware of the differences, which can be significant for relatively small counts (as observed in the chart below, the error is higher when the size of the count is small).

Another scenario that can lead to significant differences between the estimation methods described in this paper and full FPA is when there is a predominance of high or low complexity functions in the count. Counts with high complexity functions may be underestimated, while counts with low complexities will be overestimated.

The chart below shows percentual differences between estimations and FPA sizes using the data from the Nesma study.




Indicative FPA is the technique with highest errors. However, High Level FPA and SFP also show significant deviations in sizes under 1000 FPs. This may depend on

specific application or scope characteristics that can skew the results. For instance, applications with complex processes may be underestimated, while applications with simple processes may be overestimated. In these cases, such deviations could apply every time the same application is sized, for instance when sizing enhancements. A consistent introduction of the error can lead to inaccurate estimates and bad decisions down the road. Function Point analysts are therefore encouraged to analyze the differences and develop correction factors when deemed appropriate.

## Autonomous Growth and Scope Creep

Growth is the increase in function points later in the project. It may occur due to autonomous growth or 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 understand. If a user adds requirements, it is a change, the size of the product actually changes, function points are added, and the impact on 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 | |
|---|---|---|---|---|
| **Proposal** | 100 | 20% | 120 | Please note: these % values are **for illustration purposes only**. |
| **Requirements** | 109 | 10% | 120 | |
| **Design** | 115 | 5% | 120 | |
| **Construction** | 120 | 0% | 120 | |
| **Delivery & Maintenance** | 120 | 0% | 120 | |

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 20% extra function points that were already included in the requirements but could not be "seen" in that early stage of the project. The forecasted functional size of the product is 120 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 + 20%) x 10 hours / FP = 1200 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 and SFP methods can be used to size an application early in the software development life cycle. By averaging the estimates deviation and considering the effect of autonomous growth on each phase, a fairly stable function point size can be obtained and used for cost/schedule estimation. This outcome closely approximates detailed FPA, particularly on large-scale projects, while requiring significantly less time, thereby enhancing acceptance of IFPUG FPA for sizing and cost estimation.

For the final accounting, the FP calculation should be performed in detail, using FPA, since all the calculation elements are known, the requirements have been fully implemented, and the financial compensation can be calculated with the necessary precision. Furthermore, the detailed measurement, which becomes the baseline for the implemented application, is also useful for any subsequent evolutionary maintenance interventions, whether they will be estimated using one of the approximate methods or with detailed FPA, ensuring greater accuracy and consistency with reality.

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.

The following tables summarizes the comparison of the three methods presented in this document versus the detailed FPA.

| **Method** | **Accuracy** | **Cost / time to perform** | **Identify all data functions and transactions** | **Use full CPM rules** | **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 20%, depending on phase | Eventually set aside extra budget for extra functionality |
| **High Level FPA** | Medium | Medium/ Low | Yes (with assumptions) | No | Needs accounting for – e.g., add 20%, depending on phase | Eventually set aside extra budget for extra functionality |

| Method | Accuracy | Cost / time to perform | Identify all data functions and transactions | Use full CPM rules | Autonomous Growth | Scope Creep (see [1], part 2, page 4-4) |
|---|---|---|---|---|---|---|
| **SFP** | Medium | Medium/ Low | Yes (with assumptions) | No | Needs accounting for – e.g., add 20%, depending on phase | Eventually set aside extra budget for extra functionality |
| **Indicative FPA** | Medium/ Low | Low | Data: yes<br><br>Transactions: No | No | Needs accounting for – e.g., add 20%, 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?

   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. Or Use SFPs which sizes transactions and logical files independently from their 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, both for high-level FPA and SFPs!

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

   No. The high level FP estimate and SFP methods are approximations based on average complexities. 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 or SFP measures?

   They could be used when estimating contract economics and planning resource allocations, but for correct and fair economics, an FPA size should be used. An

economics review phase after delivery, based on a detailed FPA size, could be recommended.

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

   It depends on the purpose of the count. If an accurate application size is needed, detailed FPA must be used. When a rough idea of application size is enough, for instance when allocating resources, estimation methods could be used.

8. 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. In order to do that, please follow the ISO/IEC 15939 measurement process for more details.

9. 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] or apply the ISO/IEC 15939 measurement process for additional guidance.

10. What are the benefits of a detailed FPA?

    A detailed function point analysis provides a more complete definition of functional software requirements and a more accurate functional size that should facilitate estimation, 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<br>ISBN 978-0-9753783-4-2<br>(https://ifpug.mclms.net/en/package/9832/course/18997/view) |
| 2. | Early Function Point Analysis | NESMA publication<br>(https://nesma.org/wp-content/uploads/2015/11/Early-Function-Point-Analysis-vs-2015-07-15-EN.pdf) |
| 3. | The application of Function Point Analysis in the early phases of the application life cycle | NESMA publication<br>ISBN: 978-90-76258-20-1<br>(https://nesma.org/downloads/fpa-early-phases/) |
| 4. | The IFPUG Guide to IT and Software Measurement | IFPUG Publication<br>ISBN 978-1-4398-6930-7 |
| 5. | Simple Function Point (SFP) Counting Practices Manual Release v2.2 | IFPUG Publication<br>ISBN 978-0-9903007-8-6<br>(https://ifpug.mclms.net/en/package/15188/course/28227/view) |
| 6. | Simple Function Points vs Detailed Function Points Correlation | FATTOCS Publication<br>(https://www.fattocs.com/blog/function-points-vs-simple-function-points/) |

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.