Art of Software Engineering

Function Point Analysis Examined

"The bearing of a child takes nine months, no matter how many women are assigned" - Fredrick P.Brooks, Mythical Manmonth

insight

Naveen Gunti | September 2006

avenue a razorfish

Accurately predicting the size of software has always troubled the software industry. Many projects either fail or take longer than projected because the estimation was incorrect. This paper examines the technique of function point analysis (FPA) to enable the reader to understand its depth and ease as a mode of analysis, while explaining its benefits as compared to other techniques, like counting lines of code. Estimating size of software correctly is critical to the success of a project because adding more people at a later stage can make matters worse.

Problem:

Inaccurately predicting the size of a software project results in failure or in increasing the project's length.

Solution:

Function point analysis predicts software size and, therefore, project length with accuracy.

Benefits:

Function point analysis not only predicts development time but helps organizations develop expertise, which can be used for benchmarking and continuous improvement.

Lines of code is a technology-bound measure. The number of lines of code it takes to deliver a product varies based on the programming language used and has no meaning in the case of object-oriented development where everything is treated in terms of objects and classes. Allen Albrecht of IBM developed FPA as an alternative to lines of code as measure of application software size from "end user" point of view. Since an object is a true representation of data and functionality as is a function point, FPA remains more relevant for object-oriented software development.

> ***"Measuring software productivity by lines of code is like measuring progress on an airplane by how much it weighs."***
>
> ***– Bill Gates, Microsoft***

FPA has the following advantage over the lines of code technique:

- Technology independent with consistent results throughout diverse environments.
- Estimations possible early in the life cycle.
- Better communication for contract negotiations.
- Helps monitor scope creep.

What are Function Points?

Function points measure the information processing content of software systems.

- Function points measure the size of an application from the customer's point of view.
- Function points minimize overhead of the measurement process, and “keep it simple.”

One caution – function points do not measure everything, including the value of the system to the customer. A software system may have only a few function points, but many factors such as technical requirements, use of new technology, or an unsettled customer environment will cause the system to require greater than normal effort and time.

The aspects of a software system that can be measured accurately are these:

1. Inputs to the application.
2. Outputs from the application.
3. Inquiries by the end users.
4. Data files updated by the application.
5. The interface to other applications.

FPA Process Overview

The FPA process involves four elements:

1. Identifying the function point counting boundary.

   A boundary indicates the border between the software system being measured and the external application or the user domain. A boundary determines what functions are included in the function point count.

2. Determining the unadjusted function point count (UFPC).

   The unadjusted function point count reflects the specific countable functionality provided to the user by the application. The total UFPC is
   a. Data function types **(+).**
   b. Transactional function types.

3. Determining value adjustment factor (VAF).

   The value adjustment factor indicates the general functionality provided to the user of the application. VAF is comprised of 14 general system characteristics (GSCs). For each GSC, the degree of influence (DI) is determined on a 0 (lowest) to 5 (highest) scale.

VAF = 0.65 + (0.01 * Total DI)

4. Calculating final adjusted function point count (FPC).

   The FPC can be calculated using

FPC = UFPC * VAF

To calculate function points, you must first make a distinction between internal logical files and external interface files.

Internal logical files (ILF)

An internal logical file (ILF) is a user identifiable group of related data maintained within the boundary of the application:

- An ILF must be a group of data that is maintained within the application and satisfies specific user requirement.

- It include backups specifically requested by the customer but not backups done as part of information services disaster recovery requirements or automatic backup facilities provided by the technology.
- Data stores (files, database table etc) that were created for technical reasons or for storage of intermediate values (e.g. sort files, summary files) are not counted.
- Extra capabilities automatically provided are not counted unless the customer specifically requests them.

External Interface Files (EIF)

An External Interface File (EIF) is a user identifiable group of logically related data maintained outside the boundary of the application. One example of an EIF is a file or table containing names of codes (e.g. a table of department codes, product codes) read by the system being counted but maintained by some other application. The following are guidelines for counting the number of EIF:

- The group of data is logical and user identifiable, and satisfies a specific user requirement.
- The group of data is referenced by the application.
- The group of data is not maintained by the application.
- The group of data is an ILF in another application.

ILF and EIF Complexity

Each ILF and EIF contributes to the unadjusted function point count (UFPC). The number of UFPCs contributed by each ILF or EIF depends upon its complexity. Complexity of an ILF or EIF depends on the number of data element types (DET) and record element types (RET).

A data element type (DET) is a unique user recognizable field on an ILF or EIF. Some guidelines for counting data element types are:

- Count all foreign keys to other ILFs or EIFs.
- Ignore fields introduced during normalization of the ILF or EIF.
- Repeating fields (identical in format and meaning), which exist to allow multiple occurrences, are counted as a single data element type.

A record element type (RET) is a user recognizable sub-group of data elements within an ILF or EIF. Normalization of ILF or EIF is likely to identify the record element types.

Example:

The data stores about the department and its employees are as follows:

Department Details - Dept id, Dept Name, Location, HOD
Employee Details (20 employees in the department) - Emp id, Emp Name, Date of Birth, Date of Joining

The count of data element types for the above data store is 8 (total number of fields)

and

The count of record element types is 2 (subgroups of data: the department and employee).

The following table displays the complexity matrix of data element types for an ILF or EIF:

Record element types	1 TO 19 DETS	20 TO 50 DETS	51 OR MORE DETS
1	Low	Low	Average
2 to 5	Low	Average	High
6 or more	Average	High	High

The following table displays the unadjusted function point count contribution by an ILF or EIF:

COMPLEXITY	UFPC ILF	UFPC EIF
Low	7	5
Average	10	7
High	15	10

The following are some of the guidelines for ILFs and EIFs:

- An ILF or EIF is counted only once in an application even if it is maintained or accessed by many processes.
- A group of data can be counted either as an ILF or as an EIF in an application.
- Consider the user's view of the data. Do not use normalized data.
- Some physical files may not be considered part of any ILF or EIF.
- Some group of data may be counted as an ILF in more than one application, if the group of data is maintained in multiple applications.

External input (EI)

An external input (EI) processes data that come from outside the application boundary. An external input is the facility provided to the customer to insert, update, and delete records (occurrences, rows) of an ILF. It may maintain one or more ILFs. For example, an external input may maintain department and employee information. The information entered will be stored in one or more ILFs. Another example may be the maintenance of system parameters, which will be used by the processes of the software system being developed. The following are some of the guidelines for identifying the external input for a system:

- Data are received from outside the application boundary.
- The input maintains at least one ILF.
- The input is the smallest business transaction as seen by the user.
- The input is comprehensive and self contained.

External input complexity

Each external input contributes to the unadjusted function point count. The number of counts contributed by each external input depends upon its complexity, which includes the number of data element types (DETs) and file types referenced (FTRs) in the external input.

A file type referenced (FTR) is an ILF maintained or read for completing the external input or an EIF read for completing it:

- Count a FTR for each ILF maintained or read by the external input.
- Count a FTR for each EIF read by the external input.

A data element type (DET) is a unique, user recognizable field maintained on an ILF by external input. The following are some of the guidelines for counting the data element types:

- Count system generated data elements (voucher number) as data element types if the user recognizes them as part of the business input.
- Multiple physical storage is counted as a single data element type.
- Do not count screen prompts, field legends, or error messages.

The following table displays the complexity matrix for an external input of data element types:

File types referenced	1 TO 4 DETS	5 TO 15 DETS	16 OR MORE DETS
1	Low	Low	Average
2	Low	Average	High
3 or more	Average	High	High

The following table displays the unaltered function point count (UFPC) contribution by an external input:

COMPLEXITY	UFPC-EI
Low	3
Average	4
High	6

External output (EO)

An external output (EO) is a process that generates data sent outside the application boundary, for example, the external output the customer views in the form of reports, messages, etc. External outputs also include the files the application generates to be used as transactions by another application. An external output may be generated using one or more ILFs or EIFs.

The following are guidelines for identifying the external output for a system:

- Data are sent outside the application boundary.
- The output is meaningful to the customer's business.
- The output is comprehensive and self contained.
- Data in the ILF or EIF is not changed by the external output.
- Count only unique (different formats and processing) external output.
- Do not count extra capabilities provided unless the customer specifically requested them.

External output complexity

Each external output contributes to the unadjusted function point count (UFPC). The number of UFPCs contributed by each external output depends on its complexity, which depends on the number of data element type (DETs) and file types referenced (FTRs) in the external output.

A file type referenced (FTR) is an ILF read for completing the external output or an EIF read for completing it.

Count a file type referenced for each ILF and EIF read by the external output.

A data element type (DET) is a unique user-recognizable field that appears on the external output. The following are guidelines for counting them:

- Fields combined for output and recognized by the user as a single field should be counted as a single data element type.
- Do not count report titles, screen identifications, column headings, or field titles.
- Do not count page numbers or date and time of printing.

The following table displays the complexity matrix for an external output for data element types:

File types referenced	1 to 5 DETS	6 to 19 DETS	20 or more DETs
1	Low	Low	Average
2 to 3	Low	Average	High
4 or more	Average	High	High

The following table displays the unaltered function point count (UFPC) contribution by an external output:

Complexity	UFPC-EI
Low	4
Average	5
High	7

External query (EQ)

An external query is a process made up of an input-output combination that results in data retrieval. It has two parts, the screen on which the customer specifies the request (search criteria) and the resulting display. Count each unique request and display combination. The external query is unique if it has a format different from other external queries in either the request or display parts, or if the customer requests processing logic different from other external queries with the same format. On an external query, the customer enters data for control purposes to direct the search.

An external query differs from an external input since it does not modify an ILF. Though it reflects the immediate retrieval of current data for display, it differs from external output in that external output reflects the manipulation and reformatting of data (usually in report form). The media (screen or paper) is not the basis for distinguishing external queries from external output since external output can also be displayed on a terminal. An external output may be generated using one or more ILFs or EIFs.

The following are guidelines for identifying the external query for a system:

- The output is comprehensive, self contained and immediately required for the customer's business.
- When there is a one-to-one relationship between requests and displays, count only displays. Also, count just the displays if one request results in multiple displays. In either case, the count of the displays will equal the external query count. If several unique request panels result in the same display, count the requests instead of the display, for example, a display of customer information that results from completing a screen of name information, a screen of address information, or information about a specific purchase. In these cases there is one display but three external queries, since there are three different processes that get the same display.

External query complexity

Each external query contributes to the unadjusted function point count (UFPC). The number of UFPCs contributed by each external query depends upon the complexity of its input and output sides. The higher complexity of the two is taken as the complexity of the external query, which depends on the number of data element types (DETs) and file types referenced (FTRs) in the external query input or output.

A file type referenced (FTR) is an ILF or EIF read for completing the external query. Count the FTRs for the input side and output side separately. A data element type (DET) is a unique user-recognizable field that appears in the external query.

The following table displays the input complexity matrix for an external query:

File types referenced	1 TO 4 DETS	5 TO 15 DETS	16 OR MORE DETS
0 TO 1	Low	Low	Average
2	Low	Average	High
3 or more	Average	High	High

The following table displays the output complexity matrix for an external query:

File types referenced	1 TO 5 DETS	6 TO 19 DETS	20 OR MORE DETS
1 FTR	Low	Low	Average
2 to 3 FTR	Low	Average	High
4 or more FTR	Average	High	High

The higher complexity of input or output will be taken as the external query complexity.

The following table displays the unaltered function point count (UFPC) contribution by an external query:

COMPLEXITY	UFPC-EQ
Low	3
Average	4
High	6

Determine value adjustment factor (VAF)

The value adjustment factor is the determination of the complexity of the software application to be developed.

This factor is based on 14 general system characteristics (GSCs) and is used to adjust the unadjusted function point count (UFPC). Each GSC can have a degree of influence (DI) ranging from 0 (lowest) to 5 (highest).

The following are the four steps to calculate the value adjustment factor:

1. Evaluate the degree of influence (DI) of the 14 general system characteristics (GSC, which are found below) on a 0 to 5 scale.
2. Sum up the DI of all 14 GSCs to produce the total degree of influence (TDI).
   - Maximum value of TDI : 70.
   - Minimum value of TDI : 0.
   - Typical value of TDI : 35.
3. Calculate value adjustment factor using the following formula: VAF = (TDI * 0.01) + 0.65
   - Maximum value of VAF : 1.35.
   - Minimum value of VAF : 0.65.
   - Typical value of VAF : 1.00.
4. The following are the rules for DI calculation:
   - Apply on each GSC.
   - Scale from 0 to 5:
     0 - Not present, or no influence.
     1 - Incidental influence.
     2 - Moderate influence.
     3 - Average influence.
     4 - Significant influence.
     5 - Strong influence.

The following is the list of the 14 GSCs:

1. Data communication.
2. Distributed data processing.
3. Performance (response time).
4. Heavily used configuration.
5. Transaction rate (peak transaction period monthly, weekly, daily, etc.).
6. On-line data entry (what percent of total application development is data entry screens).
7. End user efficiency (degree of user friendliness provided in the application).
8. On-line update.
9. Complex processing (degree of mathematical and logical processing).
10. Reusability (What percent of code is reusable).
11. Installation ease.
12. Operational ease.
13. Multiple sites.
14. Facilitate change (flexible queries and report facilities).

Calculate final adjusted function point count

The final adjusted function point count (FPC) is calculated using the following formula:

FPC = VAF * Total UFPC

Where Total UFPC = UFPC of ILFs + UFPC of EIFs + UFPC of all EI
+ UFPC of all EO + UFPC of all EQ

Conclusion

Function points are becoming widely accepted as the standard metric for measuring software size. Now that they have made adequate sizing possible, the overall rate of progress in software productivity and software quality should improve. Understanding software size is the key to understanding both productivity and quality. Without a reliable sizing metric, relative changes in productivity (function points per person month) or relative changes in quality (defects per function point) cannot be calculated. If relative changes in productivity and quality can be calculated and studied over time, then focus can be put on an organization's strengths and weaknesses. Most importantly, any attempt for improvement can be measured for its effectiveness by putting function points to best use.

As organizations employ function points, they develop expertise, elicit accurate data, build a good repository of historic projects, and, in turn use, the data for effective benchmarking and continuous improvement.

References

http://www.sei.cmu.edu/str/descriptions/fpa_body.html

http://www.ifpug.org/about/about.htm

http://en.wikipedia.org/wiki/Function_point_analysis

About the Author

Naveen Gunti, an IT Consultant for last 13 years, using Object Oriented technologies for the last 10 years, received his Bachelors in Engineering in Electronics and Telecommunications in 1993, Masters in Systems Engineering with Honors in 1995 and Post-Graduate Diploma in Business and Industrial Management in 1996. His interests include developing artificial intelligence tools for software engineering, expert systems and neural networks, distributed computing and messaging systems. He has provided services to Fortune 500 companies in the financial, pharmaceutical, and telecommunication industries in Asia, Europe and North America.

Please share your insights and thoughts with Naveen at naveen.gunti@avenuea-razorfish.com

About Avenue A | Razorfish

Avenue A | Razorfish (www.avenuea-razorfish.com) is the largest interactive marketing and technology services firm in the U.S., and an operating unit of Seattle-based aQuantive, Inc. Avenue A | Razorfish solutions are entrenched in deep technology, rigorous analytics and a rich understanding of customer needs, including award-winning web media and creative, search marketing services, email marketing/eCRM, and world-class creative, design and implementation of customer websites, intranets and extranets. Avenue A | Razorfish operates three U.S. regions – East, West and Central – with offices located in major U.S. markets, DNA in the U.K., Amnesia in Australia and headquarters in Seattle. Clients include AstraZeneca, Best Buy, Disney, Kraft, Microsoft and Starwood Hotels & Resorts. aQuantive, Inc. and all of its operating units are committed to Internet privacy.

Avenue A | Razorfish
821 2nd Avenue, Suite 1800
Seattle, WA 98104
Phone: 206.816.8800
Fax: 206.816.8808

For more information please visit: avenuea-razorfish.com.