Q/P MANAGEMENT GROUP, INC.

The Challenges of Short Cut FPA Techniques

Roger Heller
Q/P Management Group, Inc.
10 Bow Street
Stoneham, MA 02180
www.qpmg.com

Q/P MANAGEMENT GROUP, INC.

The Challenges of Short Cut FPA Techniques

Agenda

- Description of alternative methods
  - Backfiring
  - Calibrated Backfiring
  - Estimates based on maintenance productivity rates
  - The “One File” method
  - The “Average Complexity” approach
  - Automated tools
- Benefits of doing it the old fashion way

Disclaimer

- This presentation is not intended to promote any shortcut methods to establishing function point counts
- The intent is to present a subset of the alternatives used in the industry
- The focus of this presentation is on the approach, our findings through application of the techniques and recommendations on when and if to use the technique

Backfired Function Point Estimate

Approach

- Based on the assumption that it requires a certain number of lines of code to produce a function point (LOC/FP)
- Assumes languages have different powers that equate to different LOC/FP ratios
- Public domain information created by Capers Jones is frequently quoted

Language	Level	LOC/FP
Cobol II	3	107
C++	6	53
PowerBuilder	20	16

Example: Application X has 10,000 lines of C++
Estimated FPs = 10,000 ÷ 53
= 189

Backfired Function Point Estimate (Continued)

Drawbacks to this approach:

- Counting lines of code can be difficult and sometimes impossible
- Difficult to apply when multiple languages are used
- Difficult to define what a line of code equates to in newer languages
- Varies by coding techniques
- Public domain LOC/FP ratios can be widely different than actual LOC ratios
- Margin of error could be significant (+/- 1000%)

When to use this approach:

- A quick swag is needed to:
  - Estimate the gross size of an application or application portfolio
  - Estimate how long an actual counting session should take

Calibrated Backfired Function Point Estimate

Approach

- Based on the same concepts of backfiring
- Assumes each organization would have specific LOC/FP ratios
- Calibrated LOC/FP ratios
  - Map applications to processing platform
  - Count LOC for all applications
  - Establish function point counts for 25% or more of the applications
    - Sample should be based on age and processing platform and languages
  - Create LOC/FP ratios for age groups, platform and languages
- Statistically validate predicting LOC/FP ratios to aid in degree of confidence
  - Eliminate outliers as necessary

Sample Application Function Point Estimates Using Calibrated Backfire Technique

	Age Group 1	Age Group 2	Age Group 3
Mean LOC/FP	281	448	444
STDEV	112	185	147
n	25	40	22
Confidence Level	**% variance at confidence level**	**% variance at confidence level**	**% variance at confidence level**
99%	36%	32%	45%
95%	**28%**	**24%**	**34%**
90%	23%	21%	29%
85%	20%	18%	25%
80%	**18%**	**16%**	**23%**
75%	16%	14%	20%
70%	15%	13%	18%
65%	13%	12%	16%
60%	12%	10%	15%

How to read this: For Age Group 1, there is a 95% confidence level that the accuracy of the estimate will be within ± 28% of an actual function point count. There is an 80% confidence level that the accuracy of the estimate will be within ± 18% of an actual function point count.

Calibrated Backfired Function Point Estimate
(Continued)

Drawbacks to this approach:

- Counting lines of code can be difficult and sometimes impossible
- Difficult to define what a line of code equates to in newer languages
- Varies by coding techniques
- Margin of error does exist (For our example +/- 12% to +/- 45% depending on the confidence level)
- Not an accurate estimate of a single application

When to use:

- To estimate the gross size of an application portfolio

Maintenance Productivity Rate Function Point Estimate

Approach

- Use the number of FPs one Full Time Equivalent staff (FP/FTE) can maintain as the estimating ratio
- Establish Maintenance Productivity ratios
  - Map applications to processing platform
  - Determine the number of maintenance staff required by each application
  - Establish function point counts for 25% or more of the applications
    - Sample should be based on age and processing platform
  - Create maintenance productivity ratios for age groups and processing platform using the counted applications and associated maintenance headcount
- Statistically validate predicting maintenance productivity ratios to aid in degree of confidence
  - Eliminate outliers as necessary

Maintenance Productivity Rate Function Point Estimate (Continued)

Example:

Maintenance headcount = 2.5
Maintenance Productivity for age and platform = 1,200 FPs per FTE
Estimated FPs = 1,200 * 2.5
= 3,000

Drawbacks to this approach:

- Requires accurate maintenance staff accounting by application
- Assumes applications of like age and processing platform have similar maintenance needs
- Cannot estimate the size of applications with zero support
- Margin of error does exist – similar to the calibrated backfired approach
- Cannot be used for projects

When to use:

- To estimate the gross size of an application portfolio
- To estimate the gross size of an application

The "One File" Function Point Estimate

Approach

- Assumes a fixed number of function points per logical file
  - Each file will have some number of EIs, EOs and EQs
- To determine estimated size, identify the logical files contained in the application and apply a multiplier
- Different methods apply different multipliers
  - Multiply Internal Logical Files by 31
  - Multiply Internal Logical Files by 35 and External Interface Files by 15

Example 1: Application X has 11 ILFs
Estimated FPs = 11 * 31
= 341

Example 2: Application Y has 11 ILFs and 5 EIFs
Estimated FPs = (11 * 35) + (5 * 15)
= 460

The "One File" Function Point Estimate (Continued)

Sample 1: ILF * 31

Platform	Total Counted FPs	Total Estimated FPs	Over all Delta	Average Delta	Standard Deviation	Occurrences
All Platforms	182,270	78,988	-56.66%	-33.64%	51.47%	119
Mainframe	111,475	43,834	-60.68%	-44.79%	50.78%	41
Client Server	41,585	21,824	-47.52%	-22.57%	58.57%	51
Mixed	9,711	4,774	-50.84%	-47.33%	38.49%	7
Web	19,499	8,556	-56.12%	-34.22%	29.36%	20

Sample 2: ILF * 35 & EIF * 15

Platform	Total Counted FPs	Total Estimated FPs	Over all Delta	Average Delta	Standard Deviation	Occurrences
All Platforms	182,270	102,695	-43.66%	-9.64%	64.28%	119
Mainframe	111,475	56,405	-49.40%	-19.56%	69.69%	41
Client Server	41,585	28,765	-30.83%	4.42%	68.61%	51
Mixed	9,711	6,320	-34.92%	-28.47%	55.77%	7
Web	19,499	11,205	-42.54%	-18.58%	33.33%	20

The "One File" Function Point Estimate (Continued)

Drawbacks to this approach:

- The assumption that a file supports some fixed number of transaction functions is presumptuous
- The actual FP to logical file ratio is likely to vary by platform as well as industry and application type
- Requires that every application be analyzed to identify logical files

When to use:

- When a quick rough estimate is needed
- If used, ratios should be validated with actual function point counts within the organization

The "Average Complexity" Function Point Estimate

Approach

- Assumption that on average, all functions are of "average" complexity
- A CFPS conducts a function point count
- Assume all functions are of average complexity

Sample Results

Total Counted FPs	Total Estimated FPs	Average Difference	Standard Deviation	Margin of Error - Lower	Margin of Error - Higher	Occurrences
231,848	221,829	5.55%	16.07%	2.22%	8.88%	92

The "Average Complexity" Function Point Estimate (Continued)

Drawbacks to this approach:

- To achieve an accurate count a CFPS should conduct the count
- Requires a function point count be completed but the critical complexity details are excluded
- Could result in missed functionality from lack of rigor
- Does not significantly reduce the amount of time required

When to use:

- When time is very tight and there is no need for functional complexity details

Function Point Estimating with Automated Tools

Approach

- Evaluated three different tools counting 12 applications
- Tools attempt to establish function point counts from logical data model
- Set-up is essentially the same for each tool
  - Meet with DBA and/or applications expert
  - Establish logical view of files within the application
  - Set-up tool to recognize files
- Evaluation of tools results were compared to a function point count of the same application completed by a Certified Function Point Specialist

Function Point Estimating with Automated Tools
(Continued)

Drawbacks to this approach:

- On average the counts were 300% higher than the manual counts
- The expected time savings through automated counting did not materialize
- In one instance the automated count was almost 800% higher than the manual count
- In another case the automated count was almost 400% lower than the manual count
- One count resulted in a 10% variance. The details varied significantly
  - The automated tool inaccurately identified over 150 files.
  - Files were offset by the identification of over 300 inputs, outputs and inquiries
- The code analysis tools are intended for application counting. Project counting is more difficult or impossible to perform.
- On average it took two thirds of the manual counting time to utilize the tool. This does not include time to review and adjust the count.

When to use:

- Two of the three vendors have gone out of business
- The remaining vendor does not actively promote this feature of the tool

Benefits of Doing it the Old Fashion Way

- Function point counting does not have to be time consuming
  - Experienced function point counters can accurately and efficiently count 800 – 1,500 function points per day
- Using the most accurate short cut method still requires an experienced CFPS to analyze the application and document the functions in order to achieve an inaccurate function point count
- IFPUG method is applicable to projects and applications
- IFPUG function points provide for accurate benchmarking against industry standards