Get to the Point. What’s the Deal with Different Function Points Methodologies?

A Preliminary Empirical Comparison
Authors and Collaborators

Katharine Mann
Katharine.Mann
@hq.dhs.gov

Paul Cymerman
{Pcymerman,
JVanDyke}@quaternion-
consulting.com

Ian Brown
Ibrown@avantus
federal.com

Dave Seaver
David.P.Seaver
.civ@mail.mil

Kevin McKeel
Kevin.McKeel
@logapps.com

Anandi Hira
AHira@Tecolote.com

Avantus

Logapps
Presentation Agenda

- Motivation and size methods explored
- Research methodology and dataset
- Results
- Conclusions

Table of Contents

Abstract .................................................................................................................................................. 3
Introduction .......................................................................................................................................... 3
Functional Size Metrics (FSMs) ........................................................................................................ 5
  IFPUG Function Points (FPs) ........................................................................................................... 5
  Simple Function Points (SFPs) ......................................................................................................... 7
  COSMIC Function Points (CFPs) ..................................................................................................... 8
Objective Function Points (OFPs) ....................................................................................................... 9
Effective Sizing ....................................................................................................................................... 12
Research Methodology ....................................................................................................................... 12
  Methodology ..................................................................................................................................... 12
  Dataset .............................................................................................................................................. 13
Calculating the FSMs ........................................................................................................................... 14
  Objectivity of FSM Sizing ................................................................................................................. 16
Prediction Accuracy Statistics ............................................................................................................. 17
Analysis Results ..................................................................................................................................... 18
  Comparing FSMs against Effort ......................................................................................................... 18
  Using the Objective Function Points (OFPs) Methodology ............................................................... 29
Conclusions .......................................................................................................................................... 33
Future Research ................................................................................................................................... 34
Acknowledgments ............................................................................................................................... 35
References ............................................................................................................................................ 35

Get to the Point (paper) – table of contents
MOTIVATION AND SIZE
METHODOLOGIES EXPLORED
# Software Size Metrics

<table>
<thead>
<tr>
<th>Source Lines of Code (SLOC)</th>
<th>Function Points</th>
<th>Agile Metrics (Story Points, T-shirt sizes)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pros</strong></td>
<td><strong>Pros</strong></td>
<td><strong>Pros</strong></td>
</tr>
<tr>
<td>Objective</td>
<td>Objective</td>
<td>Easy to calculate early in lifecycle</td>
</tr>
<tr>
<td>Easy to calculate at completion</td>
<td>Easier to calculate early in lifecycle</td>
<td></td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td><strong>Cons</strong></td>
<td><strong>Cons</strong></td>
</tr>
<tr>
<td>Difficult to estimate</td>
<td>Tedious to calculate</td>
<td>Highly subjective</td>
</tr>
<tr>
<td>Agile programs moving away from SLOC</td>
<td>Difficult to get actual sizes at project completion</td>
<td>Team-dependent</td>
</tr>
</tbody>
</table>
IFPUG Function Points (FPs)

- **EI**: External Input
- **EO**: External Output
- **EQ**: External Queries
- **ILF**: Internal Logical File or “Internal storage”
- **EIF**: External Interface File or “external data”

![Diagram showing IFPUG Function Points](image)
Issues with FPs

Tedious - Start

Calculating FPs requires:
• Identifying all functional transactions
• Determining correct complexity levels for each

Solution: Simple Function Points

Granularity

Transactions are limited to low, average, and high complexities.
• Very Low and Low get same sizes
• Very High and High get same sizes

Solution: COSMIC Function Points

Tedious - End

• Requirements, architecture, etc. documentation don’t match implemented solution.
• Getting actual sizes requires updating doc’s

Solution: Objective Function Points
Effective Sizing

Standard Sizing

- IFPUG and COSMIC have methods to size enhancements: sizes of the changed functional processes
  - Does not account for amount of change required (% redesign, recode, retest)
  - Does not make the modified size equivalent to new development size

Effective Sizing

- Multiply FPs with weighted average of rework %’s

Weights:

<table>
<thead>
<tr>
<th>% requirements</th>
<th>Cadence /NSA</th>
<th>Ian Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>% redesign</td>
<td>30%</td>
<td>40%</td>
</tr>
<tr>
<td>% recode</td>
<td>30%</td>
<td>25%</td>
</tr>
<tr>
<td>% retest</td>
<td>30%</td>
<td>35%</td>
</tr>
</tbody>
</table>
RESEARCH METHODOLOGY AND DATASET
Research Methodology

**Data**
- USC

**IFPUG FPs**
- Ian Brown

**COSMIC FPs**

**Simple FPs**
- DHS, NSA

**Objective FPs**
- ODNI

**Compare**
- SMC

- Utilize data from USC (Unified Code Count (UCC) enhancement tasks)
- Determine which method describes software size better
- Use tools to calculate Simple FPs from requirements, Objective FPs from code, compare to IFPUG and COSMIC FPs
Overview
- Enhancement projects
- Code metrics tool
- Command line program
- Implemented in C++, Java
- Each project by new team
- 32 data points

Groupings
- Enhancement Type
  - Add new features/modules (9)
  - Modify existing features/modules (23)
- Complexity Levels
  - Low/Average: Language Parsers, Differencing (12)
  - Very Low: Additional Metric, Input/Output (20)
**Calculated Sizes**

<table>
<thead>
<tr>
<th>Actual Functionality</th>
<th>Requirements based on Actual Functionality</th>
<th>Actual Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anandi + Colleagues</td>
<td>Ian Brown (SME)</td>
<td>DHS</td>
</tr>
<tr>
<td>Manual Process</td>
<td>Excel</td>
<td>Cadence</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UCC-G</td>
</tr>
</tbody>
</table>

**Sample Datapoint:**

<table>
<thead>
<tr>
<th></th>
<th>CFPs_AH</th>
<th>FPS_AH</th>
<th>FPS_IB</th>
<th>EFPS_IB</th>
<th>SFPs_DHS</th>
<th>SFPs_Cad</th>
<th>ESFPs_Cad</th>
<th>EOFPs</th>
<th>EOMPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makefile Parser</td>
<td>5</td>
<td>4</td>
<td>12</td>
<td>2.49</td>
<td>20.8</td>
<td>16.2</td>
<td>4.5</td>
<td>28.44</td>
<td>4.88</td>
</tr>
</tbody>
</table>

Several size metrics due to different inputs and methods or perspectives.

* NRO provides Configuration Management (CM) for UCC-G. NGA has been backing the development of the Objective method by granting access to run UCC-G on a large SW effort which provided calibration opportunities.
What We’re Comparing

Compare to Effort

- How well do these functional size metrics correlate with effort (and therefore cost)?
- Does the loss/increase in detail used to calculate size hurt/improve effort estimates?
- Which of these methods is better/more accurate for effort estimation?
- If any, what are the drawbacks to using functional size metrics for effort estimation?

Compare to Actual Effective Sizes

- Use actual reuse %’s for CFPs_AH, FPs_AH, FPs_IB, and ESFPs_Cad
- How well does this methodology predict actual, effective functional size?
SW Estimation Life Cycle

Requirements → ... → Project start → ... → Code maturity → Project completion

Function Point calculations

Get Effort Estimates

Effort Estimation Model

Objective Function Points

Continuously and iteratively update/calibrate models after programs complete with actuals

Objective

Function Points

Continuously and iteratively update/calibrate models after programs complete with actuals
RESULTS
**FPs Variants against Effort**

- Sizes stacked with large variance in effort
  - Outputs are of same size
  - Complexity and number of algorithms differ
- **Takeaway:** lack of distribution and accounting for algorithmic complexity → low correlation

- Reduced granularity compared to IFPUG FPs caused insignificant reduction in correlations
- **Takeaway:** lack of distribution and accounting for algorithmic complexity → low correlation

- Stronger positive trend between size and effort due to higher distribution
- **Takeaway:**
  - better correlation (except for Low CPLX)
  - fewer outliers/anchor points
Objective FPs against FP Variants

- Removed 5 outliers (new code, input functionality), and Average complexity projects (only 2)
- Standard % Error: 6-15%
- **Takeaway:** Promising. Not enough data for types represented in outliers

- Lack of correlation even after removal of outliers
- Not surprising – not using similar counting methodologies
- **Takeaway:** lack of correlation due to difference in methodologies

- Lack of correlation even after removal of outliers
- Not surprising – not using similar counting methodologies
- **Takeaway:** lack of correlation due to difference in methodologies
CONCLUSIONS
1. Useful? **Yes**, but reduced granularity and algorithmic complexity are problematic
   - Grouping by project/complexity type helps

2. Simple Function Points – does the loss in granularity reduce effectiveness? **No**, not in this case

3. COSMIC Function Points – does increase in granularity increase effectiveness? **Yes**, except for the Low complexity group

4. Which is the best method?
   - **COSMIC** has the highest level of granularity
   - Automated counting from requirements for **Simple** Function Points simplifies estimation process
Can the Objective Function Points method estimate actual functional size?

- Group by complexity levels, and remove projects not reusing code or creating/modifying input options ← may need more exploration
- Standard % Error for IFPUG between 6-15%
- Lack of trend for Simple and COSMIC
- Could be due to UCC atypical for Function Points

Demonstrated the technique that would be used across a more general sample or within an organization

Objective Function Points methodology still in development phase

- Improve through exposure of different software types
Future Research

- Using Function Points methodologies for Effort Estimation
  - Continue comparing estimation effectiveness across larger, varied datasets
- Objective Function Points methodology
  - Continue calibrating the method with larger and varied software products (currently working with NGA)
  - Come up with general conversions from OFPs to FPs

Acknowledgments

Portions of this work were funded by the Air Force Space and Missile Systems Center (SMC), contract FA8802-16-F-0002, and the Office of the Director of National Intelligence (ODNI) RC&E, through the CARRS contract. The authors thank Ms Adriana Contreras, Mr. Raj Palejwala, Mr. Jim Fiume, and Ms Michal Bohn for their support of this effort.