

# Automated Software Sizing is Useful Now!

---

**Michael Harris**

**David Consulting Group**

# The use of Function Points for Sizing today

---

- My perspective
  - Biggest (& best) Independent FP Analysis & Software Measurement company in the world
  - Chair of IFPUG Membership Committee
  - US & European presence, clients around the world
- Main FP Analysis Drivers
  - Outsourcing Measurement – Before & during contract
  - Benchmarking for optimization of internal sourcing e.g. comparing different product groups or geographic locations
  - Internal Process Improvement e.g. CMMI
- Who is using FPA today?
  - Some multi-nationals – especially in telecommunications
  - Increasingly government departments
  - Outsourcing vendors (being driven by their clients)
  - Not small-medium size US software developers

# IFPUG Software Tool Certification Types

---

- **Type 1 Software** provides Function Point data collection and calculation functionality, where the user performs the Function Point count manually and the software acts as a repository of the data and performs the appropriate Function Point calculations.
- **Type 2 Software** provides Function Point data collection and calculation functionality, where the user and the system/software determine the Function Point count interactively. The user answers the questions presented by the system/software and the system/software makes decisions about the count, records it and performs the appropriate calculations.
- **Type 3 Software** carries out an automatic Function Point count of an application using multiple sources of information such as the application software, database management system and stored descriptions from software design and development tools. The Software records the count and performs appropriate calculations. The user may enter some data interactively, but his or her involvement during the count is minimal. Software Type 3 instructions and criteria are currently under review by the IFPUG Board of Directors.
- The software and its associated documentation must conform to the Counting Practices Manual.

# The current situation

---

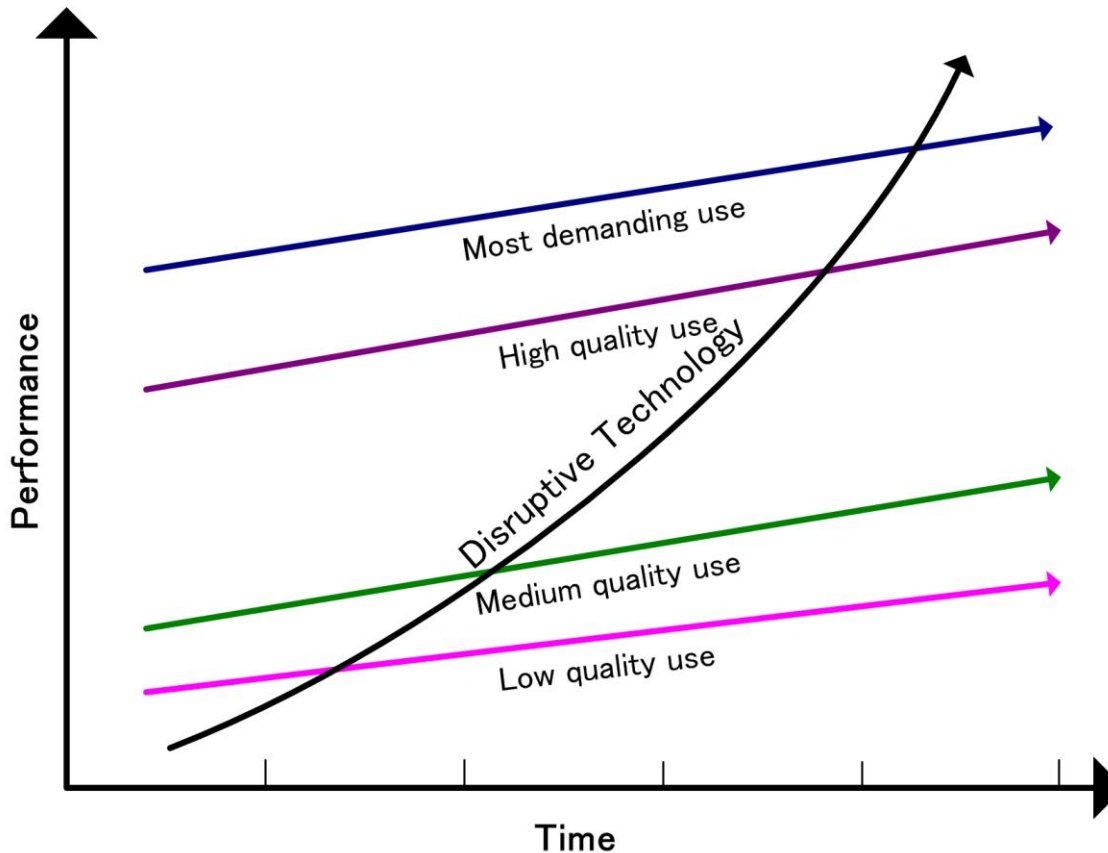
- The requirements for Type 3 are valid but represent a huge jump from Type 2 – essentially requiring a machine to count the way a human would using the same materials.
- Requirements for Type 3 may be beyond current technology



**BUT ...**

- There may be current technology that has worthwhile capabilities beyond Type 2 but nowhere near Type 3
- There are current software products that are “carefully” claiming the ability to automate FP counting.

# A Disruptive Technology?



The term *disruptive technology* was coined by Clayton M. Christensen and introduced in his 1995 article *Disruptive Technologies: Catching the Wave*, which he co-wrote with Joseph Bower. The article is aimed at managing executives who make the funding/purchasing decisions in companies rather than the research community. He describes the term further in his 1997 book *The Innovator's Dilemma*. In his sequel, *The Innovator's Solution*, Christensen replaced *disruptive technology* with the term *disruptive innovation* because he recognized that few technologies are intrinsically disruptive or sustaining in character. It is the strategy or business model that the technology enables that creates the disruptive impact. Source: Wikipedia.

# Strengths and Weaknesses of the different approaches

---

- The IFPUG approach requires the ability to deal with many different forms of input and significant pattern recognition.
- These are processes which humans are very good at.
- The subjectivity in this approach and consequent variability of human outputs is constrained (as best as it can be) by a significant body of rules – the Counting Practices Manual or CPM. This makes it time-consuming and, for some very large tasks, punitively expensive.
- A computer generally does not do subjectivity.
- Hence, if input variation can be reasonably constrained and if a reasonable set of rules can be combined into an algorithm, a computer will always produce the same result – consistently and inexpensively.
- Consequently, automation will always work better on some types of problems than others until the problem can be reformatted to suit the computer.

# Strengths and Weaknesses of the different approaches

	New Dev Project Estimate	New Dev Project @ completion	Enh. Project Estimate	Enh. Project @ completion	Application Count	Application re-count	Portfolio Baseline	Portfolio re-baseline
FPA by CFPS	Good	Good	Good	Good	Good	Good but expensive	Good but prohibitively expensive	Good but prohibitively expensive
Projection based on sample FPA by CFPS	N/A	N/A	N/A	N/A	N/A	N/A	OK but sample-sensitive	OK but sample-sensitive
Tool-supported FPA by CFPS	N/A	N/A	N/A	May be feasible	OK? – probably not less expensive	OK? – less expensive	OK? – may be less expensive	OK? – less expensive
Tool-only FPA	Not enough AI capability today to judge	Not enough AI capability today to judge	Not enough AI capability today to judge	Not enough AI capability today to judge	OK? – probably not less expensive	OK? – less expensive	OK? – may be less expensive	OK? – less expensive

# The attraction of automation

---

- While major corporations understand the need for software size metrics, the cost for a comprehensive IFPUG FP counting program in large companies can be large
- Here is an example business case that came across my desk recently for projects:
  - # of projects per year between 3000 and 9000 (depends on organizational scope of sizing program and minimum size of projects include in program) – assume 6000 projects
  - Based on profile of numbers of small, medium and large projects and type of counting (detailed or FP Lite), average cost per project for FP count is ~\$1000
  - Total cost for IFPUG counting of 6,000 projects per year is \$6,000,000.
- Combination of CAST + some IFPUG counting for calibration and special sets of projects offers a much more cost effective approach

# The Challenges presented by the current situation (1)



- FP's are used in different ways to meet different needs.
  - For some, human (CFPS) intervention will be required for the foreseeable future.
  - For others, technology available today (above level 2 but below level 3 capabilities) may be a more viable way for companies to use FP's than only human intervention.
- Consistency vs "Accuracy"
  - Current Type 3 certification requires that tools apply the CPM
  - However, the CPM rules are designed to ensure Consistency (between one CFPS and the next). For tools at the Type 2+ level, once certification is granted, there is no need for concern over consistency, the software will run the same way every time.
  - There should **not** be concern over accuracy if there is consistency.

# The Challenges presented by the current situation (2)

---

- This raises challenges of certification granularity ...By technology?  
By language?
- ... and process
  - does IFPUG keep a “gold standard” set of source code and documentation to be analyzed or
  - do we ask the vendors to bring their own?
  - How many examples do we need for statistical soundness?

# A basis for moving Forward?

Supports ...	Type 1	Type 2	Type 3 (if ever produced)	Type 2a (New?)	Type 2b (New?)	Type 2c (New?)	Type 2d (New?)
Pre-project	Y	Y	Y	N	N	N	?
Post-project	Y	Y	Y	Y	N	N	?
Application	Y	Y	Y	Y	N	N	?
Results components stored in IFPUG format	Y	Y	Y	Y	Y	Y	?
Uses CPM algorithm to calculate FP's	N	N	Y	N	N	N	?
Input: Reqmts Spec.	N	N	Y	N	N	N	?
Input: Design Spec.	N	N	Y	N	N	N	?
Input: Source code	N	N	N?	Y	Y	Y	?
Input: Human CFPS	Y	Y	N	N	Y	N	?
Output: FP	Y	Y	Y	N?	Y?	N?	?
Output: AFP	Y?	Y?	Y?	Y?	Y?	Y?	?
Use for ...	All	All	All	Productivity counts & Portfolio counts	Productivity enhancement for CFPS	Portfolio counts	?

# Where are we now? Some examples of progress

---

- Major multi-national insurance group with HQ in Switzerland
  - Uses CAST primarily for knowledge transfer to outsource vendor on old, poorly documented legacy applications
  - Uses IFPUG FP's for measurement of productivity (development) and application size (maintenance)
  - Considered CAST sizing information as one input to estimating “difficult to FP count” applications in the portfolio.
- Major US-Based Telecoms company
  - Recently completed feasibility study on calibration of new CAST **project** sizing capability against IFPUG FPs (this is very new)
  - Based on the project sample, we are seeing evidence of equivalence
  - Outcome, if acted upon, is likely to be a combination of CAST and manual counting

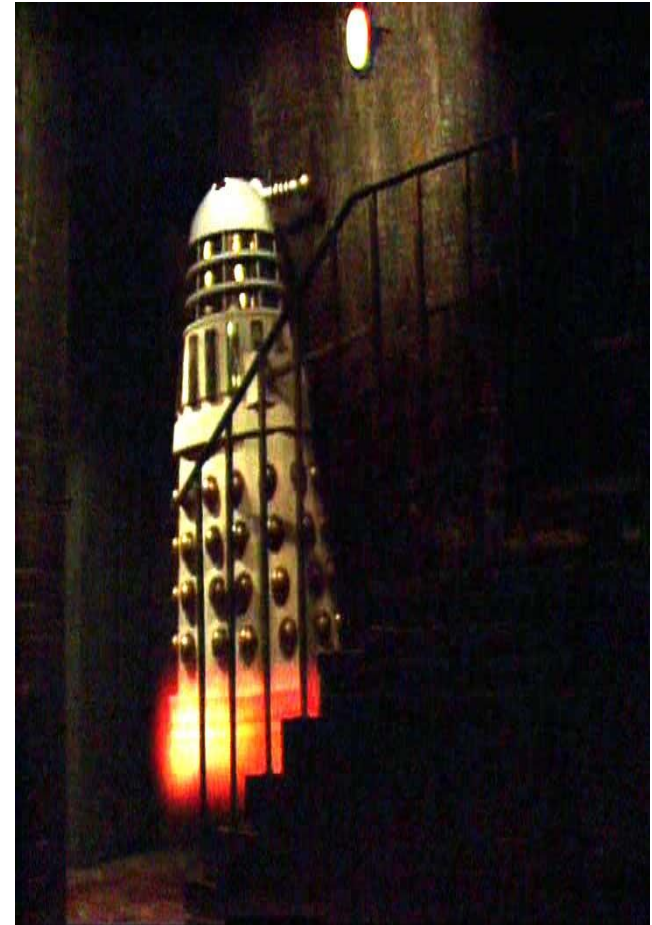
# Where are we now? Some examples of progress

---

- Major multi-national insurance group with HQ in France
  - Recently completed CAST application sizing calibration project against IFPUG FPs in one region as part of feasibility study for global roll-out
  - Outcome – repeatable process developed, in line with IFPUG Guidelines, by which CAST sizing outputs can be used to produce consistent application counts for web and mainframe

# A Way around the Challenges?

- An alternative metric?
- An alternative accreditation?
- Is there value in defining an alternative metric – perhaps Automated Function Points (AFPs) – that IFPUG (or someone else) could define using a modified version of the CPM tailored to address issues of automated tool use.
- For example, AFP's might be defined as:
  - Being traceable to standard IFPUG component elements
  - Generated from source code
  - Not including “user visibility”
  - Using simplified assumptions for data element updates by transactions



# What makes an acceptable software sizing approach?

---

- Accuracy
  - There is no absolute standard for software sizing so it is not realistic to test automated software sizing solutions for accuracy.
  - However, there are a few well-established and researched manual software sizing solutions such as IFPUG and COSMIC which can be used to give indications of comparability under different circumstances.
- Consistency
  - One of the major advantages of automated software sizing over manual sizing approaches should be the consistency of results produced by repeated sizing exercises on the same set of inputs.
- Continuity
  - Many users have long-standing software sizing programs based on publicly available approaches such as IFPUG or internal proprietary methods.
  - Often these software sizes are used for contractual commitments, service level agreements and even contractual penalties.

# What makes an acceptable software sizing approach?

---

- Proportionality
  - Software size is not a particular useful metric on its own. For a particular software sizing approach to be useful, it must have some proportionality to other software engineering artifacts such as estimated and/or actual effort to develop or maintain, defects, project duration, etc.
  - There is plenty of evidence to show that, in most cases, established manual techniques such as IFPUG exhibit proportionality.
  - If the comparison between an automated software sizing approach and IFPUG is linear (over some set of inputs), it is reasonable to expect proportionality from that automated software sizing approach.
  - There are some cases where an IFPUG software size does not exhibit proportionality. Automated software sizing solutions may claim better performance in these areas.
- Linearity
  - Related to proportionality, the relationship between the software size metric and the properties for which proportionality is desired must be linear over some worthwhile range of inputs. For example, some solutions may exhibit linearity for some types of applications but not others.

# Next “Steps”?

---

- Can “Automated FPs” be standardized?
- What is needed?
  - More discussion
  - Participation from the vendors
  - A certification framework
  - A practical approach to certification
  - A working group or sub-committee to take this and run with it.
  - Any other thoughts?
- How can you help?
  - Contact me at [m.harris@davidconsultinggroup.com](mailto:m.harris@davidconsultinggroup.com)

# Questions/Discussion?

---

Merci et au revoir!