NORTHROP GRUMMAN

DEFINING THE FUTURE

Counting Software Size: Is It as Easy as Buying A Gallon of Gas?

October 22, 2008

NDIA – 11th Annual Systems Engineering Conference

Lori Vaughan and Dean Caccavo
Northrop Grumman Mission Systems
Office of Cost Estimation and Risk Analysis

Agenda

- Introduction
- Standards and Definitions
- Sample
- Implications
- Summary

Introduction

- In what ways is software like gasoline?
- In what ways is software not like gasoline?

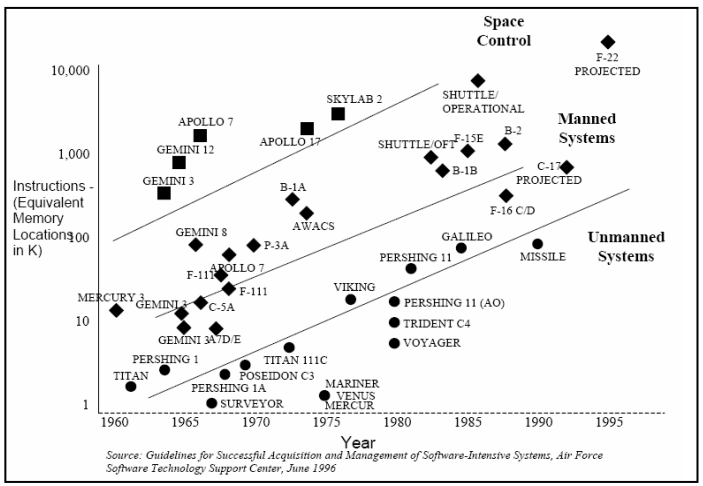
Industry Data Suggests...

- A greater percentage of the functions of the DoD Weapon Systems are performed by software

Weapon System	Year	% of Functions Performed in Software
F-4	1960	8
A-7	1964	10
F-111	1970	20
F-15	1975	35
F-16	1982	45
B-2	1990	65
F-22	2000	80

Source: PM Magazine

System Functionality Requiring Software



Code Size/Complexity Growth

- Increased amount of software in Space Systems and DoD Weapon Systems – Ground, Sea and Space/Missile
- Increased amount of software in our daily lives:
  - Cars, Cell Phones, iPod, Appliances, PDAs...

The amount of software used in DoD weapon systems has grown exponentially

Is There a Standard for Counting Software?

NORTHROP GRUMMAN
DEFINING THE FUTURE

- Since, increasing percent of our DoD systems are reliant on software we need to be able to quantify the software size
  - Historical data collection
  - Estimation and planning
  - Tracking and monitoring during program performance
- Software effort is proportional to the size of the software being developed
  - *SW Engineering Economics* 1981 by Dr. Barry Boehm
- "Counting" infers there is a standard
- Experience as a prime integrator
  - Do not see a standard being followed

There are software counting standards but the message isn't out or it is not being followed consistently

Source Line of Code definition

NORTHROP GRUMMAN
DEFINING THE FUTURE

From Wikipedia, the free encyclopedia

"**Source lines of code** (**SLOC**) is a software metric used to measure the size of a software program by counting the number of lines in the text of the program's source code. SLOC is typically used to predict the amount of effort that will be required to develop a program, as well as to estimate programming productivity or effort once the software is produced."

- Variety of Software Languages in which source code is written
  - A to Z
    - Ada, Assembler, C, C++, C#, COBOL, Fortran, Java, JavaScript, Pascal, Perl and SQL to name just a few

Source Line of Code definition: Physical and Logical

NORTHROP GRUMMAN
DEFINING THE FUTURE

- Software Engineering Institute (SEI) has developed checklist as part of a system of definition checklists to support measurement definitions [Software Size Measurement: A Framework for Counting Source Statements.]

Definition Checklist for Source Statement Counts

Definition name: *Logical Source Statements (basic definition)* Date: *8/7/92* Originator: *SEI*

Measurement unit: Physical source lines ☐ Logical source statements ✓

Statement type	Definition ✓	Data array ☐		Includes	Excludes
When a line or statement contains more than one type, classify it as the type with the highest precedence.					
1 Executable	Order of precedence ->		1	✓	
2 Nonexecutable					
3 Declarations			2	✓	
4 Compiler directives			3	✓	
5 Comments					
6 On their own lines			4		✓
7 On lines with source code			5		✓
8 Banners and nonblank spacers			6		✓
9 Blank (empty) comments			7		✓
10 Blank lines			8		✓
11					
12					

- Physical SLOC: One physical SLOC is corresponding to one line starting with the first character and ending by carriage return or an end of file marker of the same line and which excludes the blank and comment line.
- Logical SLOC: Lines of code intended to measure “statements” which normally terminated with a semicolon or a carriage return. Logical SLOC are not sensitive to format, style and conventions, but they are language dependent.

Source Line of Code Samples

NORTHROP GRUMMAN
DEFINING THE FUTURE

```
for (i=0; i<100; ++i) printf("hello"); /* How many lines of code is this? */
```

- 1 Physical Line of Code LOC
- 2 Logical Lines of Code LOC (for statement and printf statement)
- 1 Comment Line

```
for (i=0; i<100; ++i)
 {
 printf("hello");
 } /* Now how many lines of code is this? */
```

- 4 Physical Lines of Code LOC (Is placing braces work to be estimated?)
- 2 Logical Line of Code LOC (What about all the work writing non-statement lines?)
- 1 Comment Line (Tools must account for all code and comments regardless of comment placement.)

Note the logical count is independent of the programming style and conventions

Implications of SLOC Counts

NORTHROP GRUMMAN
DEFINING THE FUTURE

Typical Simplified Software Cost Estimation Formula

- Suppose you were given this simplified software cost formula and you received data from two separate contractors and were asked to determine relative development costs?
- What would that impact?
  - Size
  - Productivity
  - Hours

Implication Illustration – Historical

NORTHROP GRUMMAN
DEFINING THE FUTURE

Contractor A

Physical Coordinate Perspective

SLOC Count	500 KSLOC
Effort	2500 Person Months (PM)
Productivity	500 KSLOC ÷ 2500 PM= 200 ESLOC/PM

Contractor B

Logical Coordinate Perspective

SLOC Count	312.5 KSLOC
Effort	2500 (PM)
Productivity	312.5 KSLOC ÷ 2500 PM = 125 ESLOC/PM

Without understanding the basis of the Software SLOC count, it looks like Contractor A is more productive. Is this correct?

Implication Illustration – Estimate Comparison

NORTHROP GRUMMAN
DEFINING THE FUTURE

Contractor A

Estimated Size	600 KSLOC
Historical Productivity	200 ESLOC/PM
Estimated Effort	3,000 PM
Estimated Cost	3,000 PM X $20K = $ 60 M

Contractor B

Estimated Size	600 KSLOC
Historical Productivity	125 ESLOC/PM
Estimated Effort	4,800 PM
Estimated Cost	4,800 PM X $20K = $ 96 M

USC Center for Systems and Software Engineering

NORTHROP GRUMMAN
DEFINING THE FUTURE

- Attributes of a good code counter
  - Non Proprietary
  - Available to the public
  - Platform independent
  - Support multiple programming languages
  - Count both physical and logical SLOC
  - Limited Public License or “Copyleft” type agr
- http://sunset.usc.edu/research/CODECOUNT/

Sample 1.0::SLOC Counting
The Totals

Total Lines	Blank Lines	Comments Whole	Comments Embedded	Compiler Direct.	Data Decl.	Exec. Instr.	Number of Files	File SLOC Type	SLOC Definition
33991	3855	8465	19	250	6815	14606	336	21671	CODE Physical
33991	3855	8465	19	250	2775	10667	336	13692	CODE Logical
1135	42	0	0	0	1093	0	47	1093	DATA Physical

Number of files successfully accessed..... 383 out of 383

Ratio of Physical to Logical SLOC..... 1.58

USC CSSE CodeCount™

NORTHROP GRUMMAN
DEFINING THE FUTURE

- What programming languages are covered today
  - Ada , Assembler(s), Jovial, Pascal, COBOL, Fortran, MUL – Markup Language, Java, C/C++, C#, JavaScript, Visual Basic and Visual Basic Script
- What is included for each language
  - Read me file
  - Logical Standard (word table)
  - C source code of language specific counter
  - Sample input, source files and output file

USC Center for Systems and Software Engineering (CSSE) CodeCount™ suite supports many languages

Imagine Software Code Counting…

NORTHROP GRUMMAN
DEFINING THE FUTURE

- As an integral part of your program's change management system
- Improving your ability to perform Root cause Analysis
- Normalized code counts of existing software that are automatically uploaded to your historical database
- A historical repository of software size that could be used for estimation purposes and parametric model calibration
- Improving the representative nature of Parametric and Predictive Modeling
- Being consistent….

Summary

- Recognize underlying implications of Physical and Logical software sizing
- Assess appropriateness and magnitude of code count measurement
- Consider widespread standardization and integration into acquisition process

NORTHROP GRUMMAN

DEFINING THE FUTURE