NATIONAL RECONNAISSANCE OFFICE

Objective SLOC: An Alternative Method to Sizing Software Development Efforts

Andrew Kicinski
Integrity Applications Incorporated
NRO/Cost and Acquisition Assessment Group (CAAG)

ICEAA June 2017

SUPRA ET ULTRA

Agenda

- Software Estimating Today: the ESLOC Method
- Need for a New Approach
- ESLOC Alternatives – OSLOC (Objective SLOC) and Parametric Models
- Future of Software Estimating

- BLUF: A parametric model and an estimate by analogy approach have been developed to provide a more objective, simplified and defendable software development cost estimate

How Software Development Effort is Measured

- Level of Effort
- Function Points
- Source Lines of Code (SLOC)
- Commercial Models – SEER SEM, COCOMO, SLIM, Price

How Software Development Effort is Measured at the CAAG

- Equivalent Source Lines of Code (ESLOC)
  - Primary method of software (SW) estimating by NRO CAAG
  - A proxy for effective software development effort
  - Standardizes new and reuse code to a single effective measure
    - Assumes effort to reuse SW is less than or equal to new SW development
  - Derived from commercial standards

$$ESLOC = New + .25 \times Autogen + (Unmodified + Modified) \times \%Rework$$

where

$$\%Rework = (.4 \times \%RD) + (.25 \times \%RI) + (.35 \times \%RT)$$

$$\%RD = \%Redesign$$
$$\%RI = \%Reimplementation$$
$$\%RT = \%Retest$$

ESLOC Alternative Analysis

- The CAAG recognizes the weakness of the current ESLOC method is rooted in the subjective RD/RI/RT inputs

Partially Subjective

Subjective

$$ESLOC = New + .25 \times Autogen + (Unmodified + Modified) \times (.4 \times \%RD + .25 \times \%RI + .35 \times \%RT)$$

Objective

- The "ESLOC Alternative Analysis" study was recently implemented to assess **objective** alternatives to ESLOC
- Goals of this study were:
  - Evaluate the current ESLOC method
  - Propose and develop new objective measures for estimating effective SW size
  - Assess viability and compare performance of objective measures to ESLOC
  - Recommend path forward for CAAG SW estimating team

ESLOC Advantages

- ESLOC allows the scaling of reuse code based on the expected or observed effort to use the existing software
- Higher RD/RI/RT values should accompany more effort to utilize pre-existing code

Lower RD/RI/RT
- Internal reuse
- Non-mission critical SW
- Mature reuse baseline

Higher RD/RI/RT
- External reuse
- Mission critical SW
- Low-maturity reuse

- Example (perspective of SME populating SW datasheets):

	ITEM SIZE DATA									
	SOURCE	DELIVERED NEW CODE		PRE-EXISTING CODE						
Logical SLOC	LANGUAGE	UNIQUE SLOC	AUTO GEN SLOC	TOTAL UN-MODIFIED SLOC	TOTAL MODIFIED SLOC	TOTAL DELETED SLOC	%RD	%RI	%RT	ESLOC
32,000	C++	5,000	0	25,000	2,000	3,000	5	1	10	6,553
32,000	C++	5,000	0	25,000	2,000	3,000	10	7	30	9,365

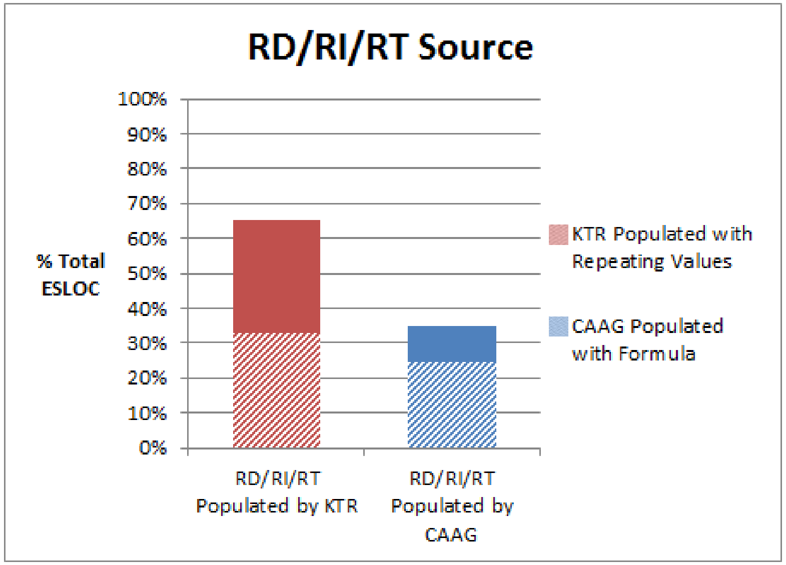
ESLOC Disadvantages

- Although well intentioned, ESLOC parameters (RD/RI/RT):
  - Need to be populated by an analyst intimately familiar with the SW
  - Are often misunderstood, misinterpreted, not populated, or populated with repeating values (same value for all SW components)
  - Can have large impact on ESLOC from small changes
  - Vary widely across programs, contributing to additional uncertainty and variability in SW productivities
  - Compound pre-existing code in cases of multiple SW snapshots
  - Cannot be independently verified – defending changes is difficult
- Example (perspective of CAAG analyst verifying SW datasheets):

	ITEM SIZE DATA									
	SOURCE	DELIVERED NEW CODE		PRE-EXISTING CODE						
Logical SLOC	LANGUAGE	UNIQUE SLOC	AUTO GEN SLOC	TOTAL UN-MODIFIED SLOC	TOTAL MODIFIED SLOC	TOTAL DELETED SLOC	%RD	%RI	%RT	ESLOC
32,000	C++	5,000	0	25,000	2,000	3,000	5	1	10	6,553
32,000	C++	5,000	0	25,000	2,000	3,000	10	7	30	9,365

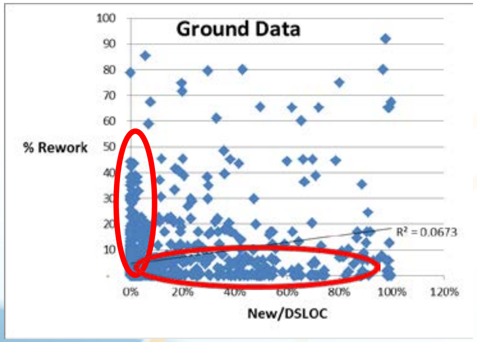
ESLOC Disadvantages Quantified

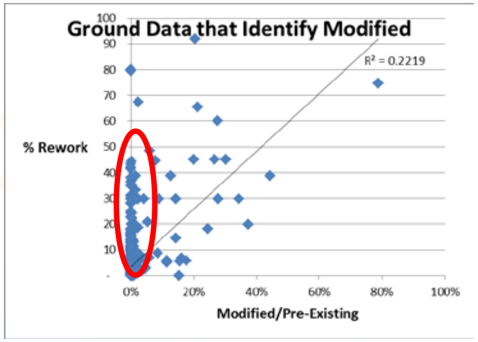
- We hypothesize ESLOC has many issues. What data backs up this claim? An all-encompassing NRO ground dataset was compiled and the following metrics were calculated:



- More than one-third of ESLOC was based on CAAG-populated RD/RI/RT
- Half of the ESLOC resulting from contractor-populated RD/RI/RT used repeating RD/RI/RT values (same values for multiple SW items)

$$\%Rework = (.4 \times \%RD) + (.25 \times \%RI) + (.35 \times \%RT)$$





- %Rework shows very little correlation to %New or %Modified
- There is significant variation, verifying low quality of subjective RD/RI/RT
  - High %New but low %Rework
  - Low %New but high %Rework
  - Low %Modified but high %Rework

ESLOC Alternatives

- The evidence is clear: ESLOC needs to be replaced

  What are the objective alternatives?

- Option 1: Set RD/RI/RT objectively
- Option 2: Assert an Objective SLOC (OSLOC) formula
- Option 3: Use regression techniques to derive CER-type method

Evaluation of Methods

- Standard model quality metrics were used to evaluate different options, including Standard Percent Error (SPE), correlation (R^2), average bias and error residual trending

- Distribution and range of productivities was also considered as a way to compare methods
  - ESLOC has a large range of productivities and is highly skewed, due to variability and uncertainty surrounding RD/RI/RT
  - Less skew and tighter range of productivities indicates less uncertainty of inputs
  - Evaluated standard deviation, skewness and 80^{th} percentile divided by 20^{th} percentile as characterizations of productivity distribution

Option 1: Set RD/RI/RT Objectively

- RD/RI/RT vary significantly due to their high subjectivity. If these values could be assigned objectively, our sizing method would contain less uncertainty
- We have observed contractors using formulas to populate RD/RI/RT and have begun internally populating %RI as %Modified when no better information is available
- Option 1a: set RD/RI/RT as the following

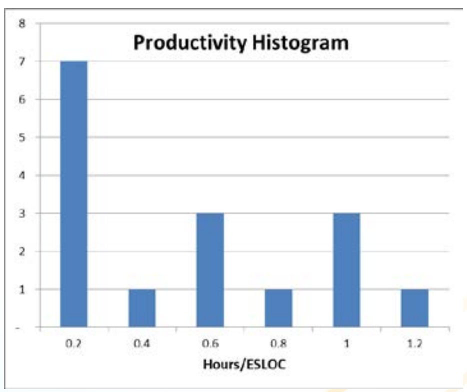
RD	RI	RT
5%	Modified/ Pre-Existing	10%

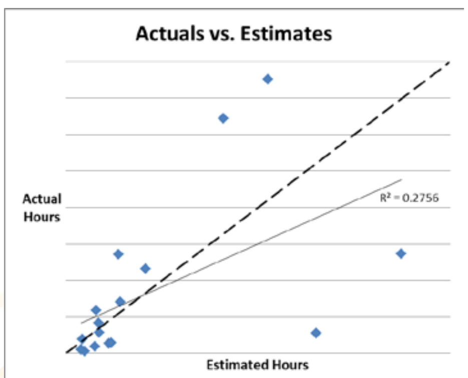
ESLOC

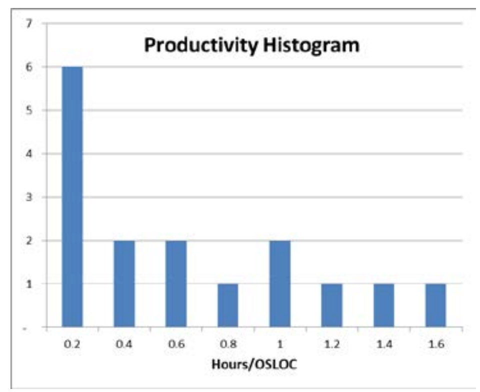
Hours/ESLOC Distribution		Model Statistics	
80th / 20th	6.09	Bias	-2%
Skew	0.43	SPE	76%
Stdev	0.36	R^2	0.28

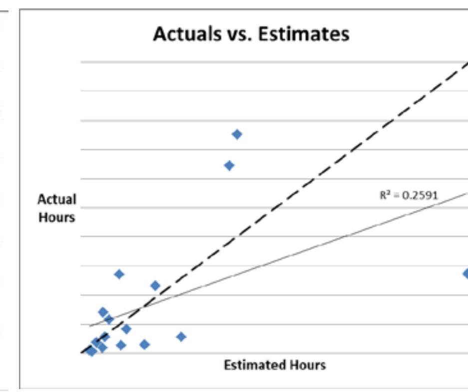
OSLOC Option 1a

Hours/OSLOC Distribution		Model Statistics	
80th / 20th	6.68	Bias	0%
Skew	0.84	SPE	84%
Stdev	0.45	R^2	0.26









*Results on subset of ground data that identify Modified SLOC

Using %Modified as %RI and using SEER standards for %RD and RT does not improve estimating method

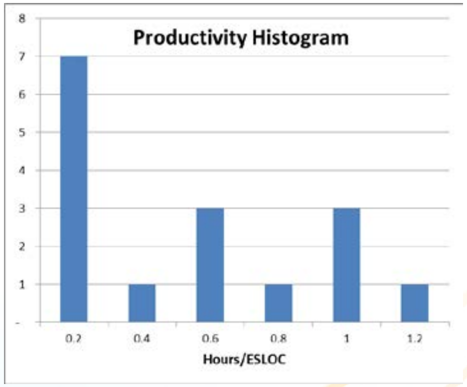
Option 1: Set RD/RI/RT Objectively

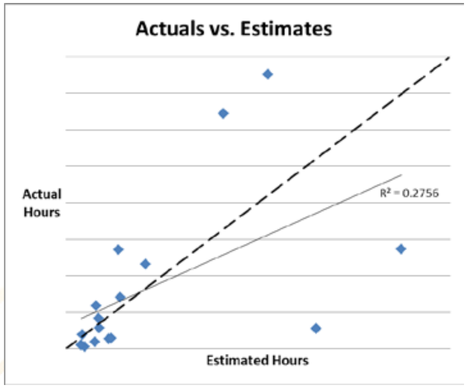
- Option 1a set RD and RT to SEER SEM standards for reuse. This standard may not be appropriate for every SW CSCI.
- Option 1b: set all of RD/RI/RT to Modified/Pre-Existing, so

$$OSLOC = New + .25 \times Autogen + (Unmod + Mod) \times \frac{Mod}{PreExisting}$$

ESLOC

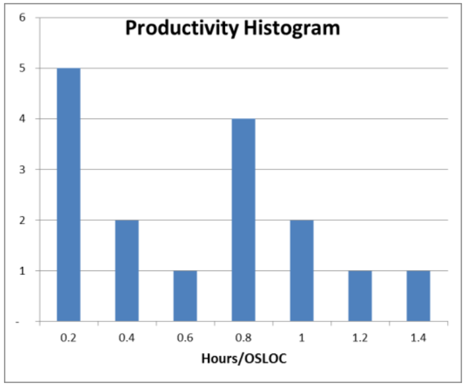
Hours/ESLOC Distribution		Model Statistics	
80th / 20th	6.09	Bias	-2%
Skew	0.43	SPE	76%
Stdev	0.36	R^2	0.28

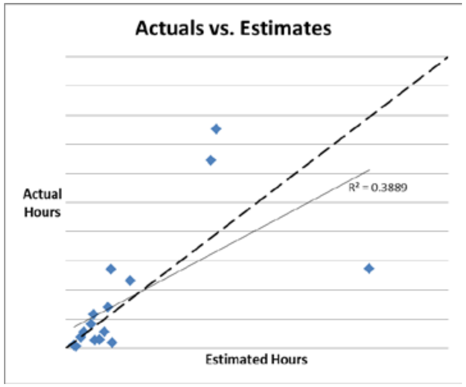




OSLOC Option 1b

Hours/OSLOC Distribution		Model Statistics	
80th / 20th	4.72	Bias	0%
Skew	0.33	SPE	67%
Stdev	0.36	R^2	0.39





*Results on subset of ground data that identify Modified SLOC

Using %Modified as the entire rework percentage provides some improvement over ESLOC

Option 2: Assert an OSLOC Formula

- Option 1b was $OSLOC = New + .25 \times Autogen + (Unmod + Mod) \times \frac{Mod}{PreExisting}$
- If Autogen is small, and not expected to be a large influencer, and since $Pre\text{-}Existing = Unmod + Mod - Deleted$, if Deleted is small then effectively,

$$OSLOC = New + (Unmod + Mod) \times \frac{Mod}{Unmod + Mod}$$

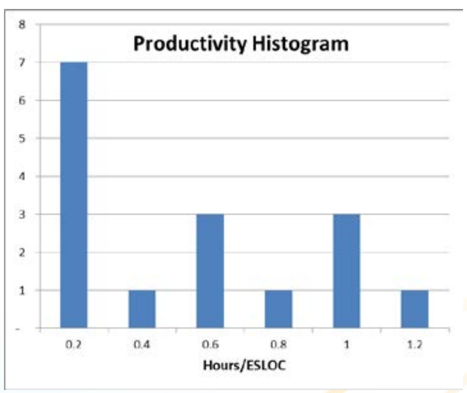
Option 2a: $OSLOC = New + Mod$

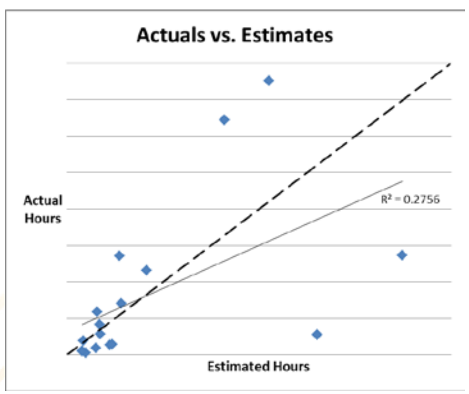
ESLOC

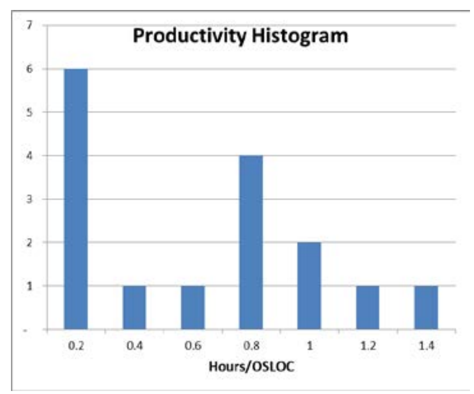
Hours/ESLOC Distribution		Model Statistics	
80th / 20th	6.09	Bias	-2%
Skew	0.43	SPE	76%
Stdev	0.36	R^2	0.28

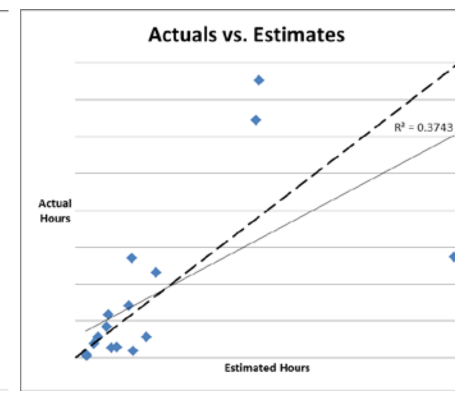
OSLOC Option 2a

Hours/OSLOC Distribution		Model Statistics	
80th / 20th	4.93	Bias	0%
Skew	0.36	SPE	69%
Stdev	0.37	R^2	0.37









*Results on subset of ground data that identify Modified SLOC

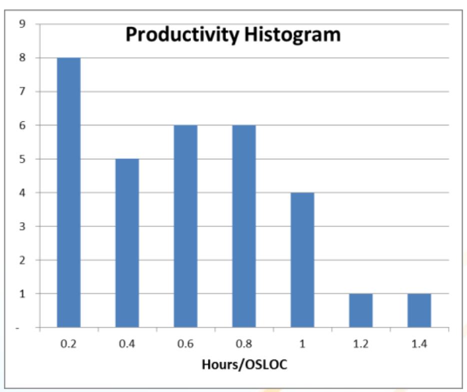
New + Modified is a simple sizing metric and performs better than ESLOC and similar to Option 1b

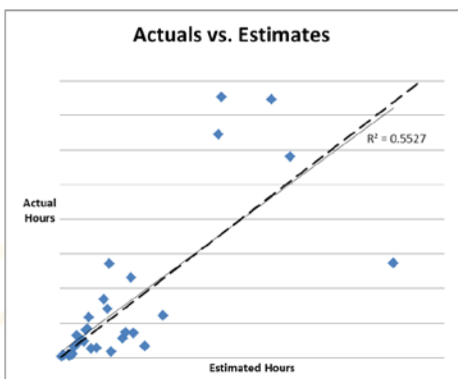
Option 2: Assert an OSLOC Formula

- Dataset includes programs of varying levels of confidence
  - Completed/on-going
  - UCC/contractor counter/estimate
  - Normalization/mappings being reassessed
  - Modified code identified/not identified
- Option 2a was run on three datasets
  1. Ground programs that identify modified (previous chart)
  2. All ground programs
  3. Ground programs that identify modified using UCC and have no significant DQ issues

2. All ground programs

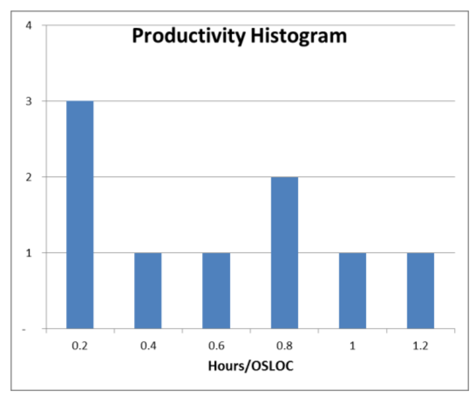
Hours/OSLOC Distribution		Model Statistics	
80 / 20	4.23	Bias	0%
Skew	0.47	SPE	64%
Stdev	0.32	R^2	0.55

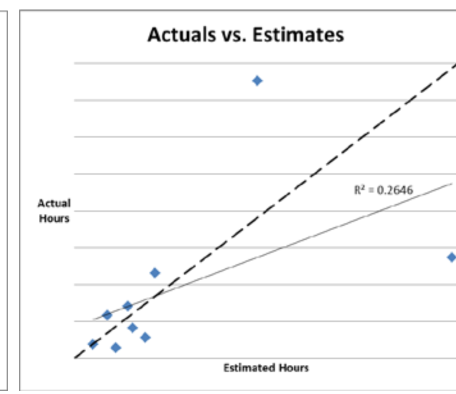




3. Ground programs that identify modified using UCC and have no significant DQ issues

Hours/OSLOC Distribution		Model Statistics	
80th / 20th	4.17	Bias	0%
Skew	0.32	SPE	62%
Stdev	0.33	R^2	0.26





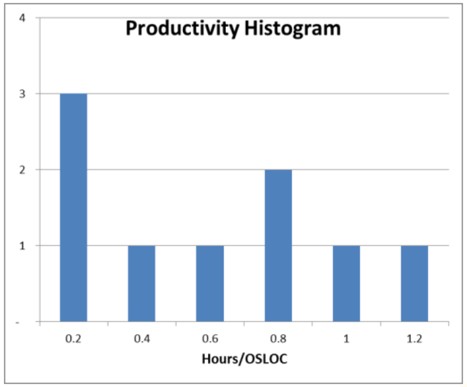
New + Mod performs similarly on a larger set including low quality data and on a small set of high quality data

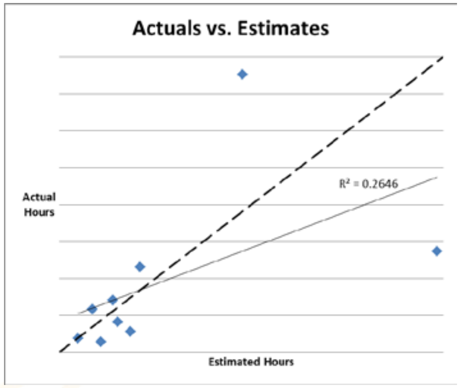
Option 2: Assert an OSLOC Formula

- Recently we have begun collecting metrics on data SLOC (XML and HTML) and have been decrementing Data ESLOC in some cases
- The effect of data SLOC was tested on New + Modified (Option 2a) on the UCC data subset by removing all New and Modified data code (Option 2b)

Option 2a

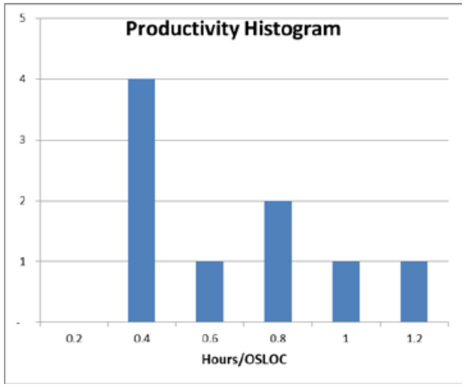
Hours/OSLOC Distribution		Model Statistics	
80th / 20th	4.17	Bias	0%
Skew	0.32	SPE	62%
Stdev	0.33	R^2	0.26

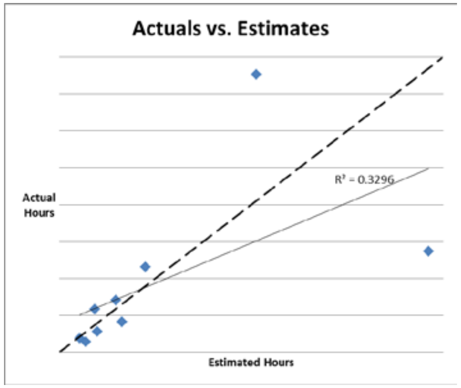




Option 2b

Hours/OSLOC Distribution		Model Statistics	
80th / 20th	2.42	Bias	0%
Skew	0.43	SPE	51%
Stdev	0.29	R^2	0.33





- Similar results show removing HTML and XML from code counts improves OSLOC model on set of all NRO ground SW programs

Removing data from OSLOC improve Standard Error and reduces range of OSLOC productivities

Option 3: Use Regression Techniques to Derive CER-type Method

- Parametric models were run to see if they could outperform a simple New + Modified OSLOC equation
- Due to the skewed distributions of New, Unmodified, Modified and Deleted SLOC, LOLS on multiplicative forms is the preferred regression method

CER Tab Name	CER Function	SPE	R^2
ZMPE ESLOC Base	SW Dev Hours = a*ESLOC	66.2%	0.45
LOLS ESLOC Base Exp	SW Dev Hours = a * ESLOC^b	69.0%	0.46
ZMPE 1	SW Dev Hours = a*New	147.1%	0.53
ZMPE 2	SW Dev Hours = a*(New+Modified)	63.5%	0.55
LOLS 3	SW Dev Hours = a * New^b	119.2%	0.56
LOLS 4	SW Dev Hours = a * (New+Mod)^b	63.6%	0.55
ZMPE 5	SW Dev Hours = a*New^b + c*Mod^d	63.3%	0.52
LOLS 6	SW Dev Hours = a*New^b * (Mod/New+1)^c	65.4%	0.55
LOLS 7	SW Dev Hours = a*New^b * (Mod/New+1)^c * (Unmod/New+1)^d	65.3%	0.74

*Results on set of all NRO ground data

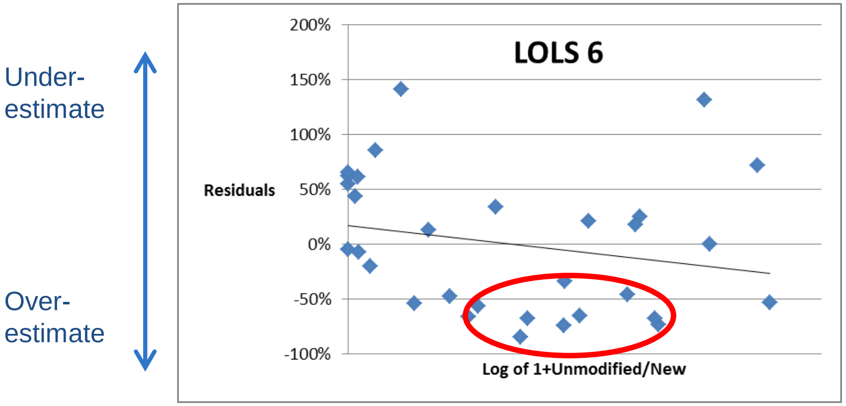
- CER models produce similar regression statistics to OSLOC models
- LOLS 7 produced a model suggesting high unmodified SLOC was associated with less effort ($d < 0$), inconsistent with expectations

Investigating Unexpected CER Behavior

- High amounts of unmodified reuse should take some additional effort to understand, integrate with new code, and retest

  What could cause a regression model to produce the opposite conclusion?

- LOLS 6: $SW\ Dev\ Hours = a \times New^{b} \times \left(1 + \frac{Mod}{New}\right)^{c}$



Multiple Differencing Example

	Baseline A	Baseline B	New	Unmod	Mod	Deleted	Pre-Existing	DSLOC
Multiple Diff	DLV 1.0	DLV 2.0	100	900	50	50	1,000	1,050
	DLV 2.0	DLV 3.0	150	950	75	25	1,050	1,175
	Sum		**250**	**1,850**	**125**	**75**		
Single Diff	DLV 1.0	DLV 3.0	**225**	**850**	**100**	**50**	1,000	1,175

Multiple differencing snapshots tend to capture more churn and have higher SLOC counts than a single diff run

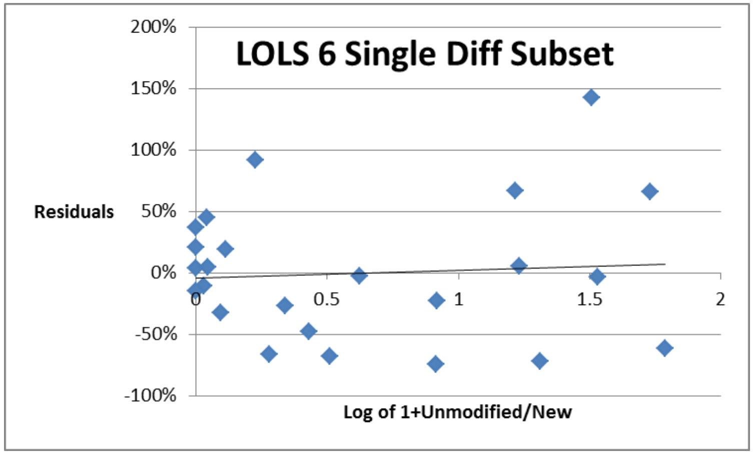
- Residual plot on LOLS 6 shows adding an unmodified scaling factor does not improve model based on expectations
  - SW programs with large amounts of unmodified SLOC are already being over-estimated
- It was discovered that six of seven data points that consisted of multiple deliveries were over-estimated and are contained within the red oval – maybe these programs are being over-estimated because of how code counts were reported

CER on Subset of Data

- Promising CER models were run on the set of ground SW programs that reported SW sizing based on one differencing run (7 DPs removed)

CER Tab Name	CER Function	SPE	R2
LOLS 6	SW Dev Hours = a*New^b * (Mod/New+1)^c	65.4%	0.55
LOLS 6 single diff subset	SW Dev Hours = a*New^b * (Mod/New+1)^c	57.7%	0.92
ZMPE 6 single diff subset	SW Dev Hours = a*New^b * (Mod/New+1)^c	51.7%	0.89

- Standard error and correlation improve significantly
- Unmodified now shows expected positive relationship, but provides very little additional explanatory power



- 7 data points composed of multiple SW deliveries have virtually nothing else in common – different contractors, ground function, size, etc. – there is no reason to believe there is another reason contributing to their previous over-estimation
- Removing XML and HTML code improves models further

CER Tab Name	CER Function	SPE	R2
LOLS 6 single diff w/o data	SW Dev Hours = a*New^b * (Mod/New+1)^c	54.2%	0.91
ZMPE 6 single diff w/o data	SW Dev Hours = a*New^b * (Mod/New+1)^c	49.0%	0.88

Option 2: OSLOC Formula – on Subset

- Removing data points that were composed of multiple SW deliveries improved the CER models

Can reducing the set to those with one SW differencing summary improve the results of the OSLOC model?

- Recall the best performing OSLOC model was Option 2b:

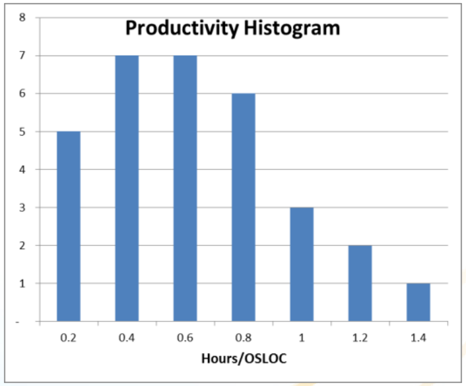
$$OSLOC = New + Modified\ (excl.\ XML, HTML)$$

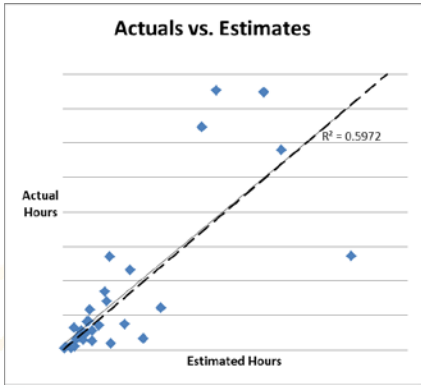
Option 2b on all NRO ground

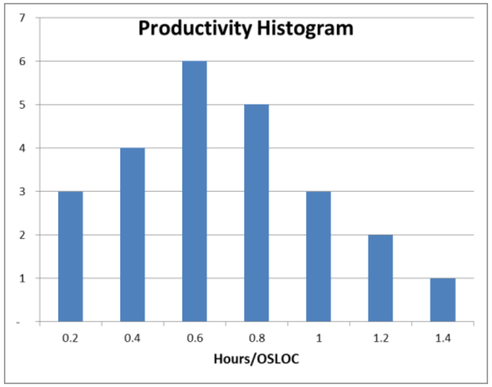
Hours/OSLOC Distribution		Model Statistics	
80 / 20	3.34	Bias	0%
Skew	0.54	SPE	61%
Stdev	0.32	R^2	0.60

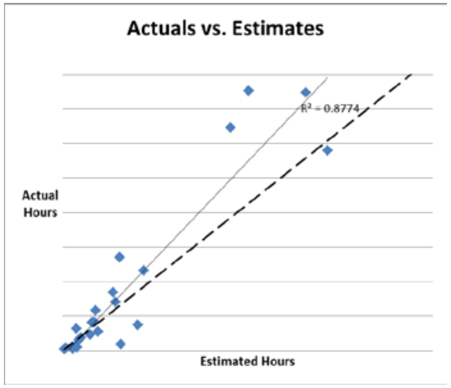
Option 2b on subset

Hours/OSLOC Distribution		Model Statistics	
80 / 20	3.31	Bias	0%
Skew	0.27	SPE	55%
Stdev	0.33	R^2	0.88









OSLOC Model improves when removing programs with multiple diffs, but does underestimate larger programs

Recommended Models

#	Model Attribute	OSLOC 2b	CER 6
1	Data collection going forward will be completely objective through the use of UCC-G	X	X
2	Simple to understand and implement	X	X
3	Reduces burden to contractor and improves CAAG ability to defend estimates	X	X
4	Performs significantly better when all data is based on a single SW differencing summary	X	X
5	Estimate by analogy (choose analogous program SW productivity)	X	
6	Estimate by parametric model (no analogy needed)		X

OSLOC 2b: $OSLOC = New + Modified\ (excl.\ XML, HTML)$

CER 6: $SW\ Dev\ Hrs = a * New^{b} * \left(1 + \frac{Mod}{New}\right)^{c}$

Best OSLOC and parametric model perform similarly and share many of the same desirable characteristics

Future of SW Estimating at CAAG

- CAAG to begin a parallel path approach to SW sizing and estimating
  - OSLOC metrics will be calculated and collected for all historic programs and future collections
  - Future estimates will investigate applying OSLOC method and parametric model as alternative methods of estimating and as cross checks
  - ESLOC metrics will be maintained and ESLOC inputs will continue to be collected to allow the analyst the option of reverting to estimate by ESLOC analogy should OSLOC and the parametric model not meet their needs
- Good practices that will be sought after to improve objective SW estimating
  - Recommend calculating SW differencing counts between the initial and current SW baselines
  - CAAG should ensure contractors always run UCC-G and run it correctly
  - Ensure documentation of software functionality exists to complement software sizing
- While OSLOC is still in "beta testing" we hope to see improvements in our ability to objectively estimate software development. Results and implementation will be reviewed and shared in the future

Questions?

Thank you!

Andrew Kicinski

kicinski@nro.mil
akicinski@integrity-apps.com

571-304-8867

NATIONAL RECONNAISSANCE OFFICE

SUPRA ET ULTRA

The ESLOC Method

- The CAAG has historically used the ESLOC method to estimate SW development
- Equivalent Source Lines of Code (ESLOC) is a standardizing measure
  - 1 new line of code = 1 ESLOC
  - 1 autogenerated line of code = .25 ESLOC
  - 1 unmodified or modified line of code ≤ 1 ESLOC
    - Reuse is scaled based on an assessment of the percent redesign, reimplementation and retest (RD/RI/RT)

$$ESLOC = New + .25 \times Autogen + (Unmodified + Modified) \times (.4 \times \%RD + .25 \times \%RI + .35 \times \%RT)$$

- How the ESLOC method applies to our processes:

Data collection process:

1. Contractor runs UCC to collect objective sizing
2. Contractor assesses rework effort and provides RD/RI/RT
3. CAAG normalizes raw data including mapping hours/costs to SW Dev
4. SW Metrics are produced (Hours/ESLOC)

Point estimate process:

1. Contractor populates SW Datasheet including SW sizing and RD/RI/RT
2. SW sizing and RD/RI/RT are assessed for reasonability and adjusted as necessary, producing ESLOC
3. Analogous program SW productivities and labor rates are pulled as assumptions
4. $ESLOC \times \frac{Hours}{ESLOC} \times \frac{\$BY}{Hour} = Point\ Est$

Evaluation of Methods

- Typically in model development, parametric models, such as CERs, can be evaluated by comparing actual costs to predicted costs by utilizing the proposed model and assessing SPE, R^2, bias, residual trending, etc.
  - This approach was taken for Option 3 (use regression techniques to derive CER-type method)
- Assessing Options 1 (set RD/RI/RT objectively), 2 (assert OSLOC formula) and the current ESLOC method are more difficult
  - In practice these methods involve estimating by analogy
  - During methods development it is difficult to apply an analogous productivity to make the actual to predicted hours comparison
  - For our assessments, it was assumed that the average data set productivity would be the applied analogy to derive predicted hours
- Distribution and range of productivities were also considered as ways to compare methods
  - ESLOC has a large range of productivities and is highly skewed, due to variability and uncertainty surrounding RD/RI/RT
  - Less skew and tighter range of productivities indicates less uncertainty of inputs
  - Evaluated standard deviation, skewness and 80th percentile divided by 20th percentile as characterizations of productivity distribution

CAAG SW Datasheet

End Item Software Datasheet 1

Preparer:		Baseline A:	
Secure Phone:		Baseline B:	
Email:		Baseline A Date:	
Company:		Baseline B Date:	
Date:			
Site:			

Instructions:
- Identify Baselines A and B that were run through the UCC differencing function to populate this Datasheet. Identify the dates of most recent update to the baselines.
- Use a new End Item SW Datasheet 1-3 for additional differencing results for other baselines.
- Use UCC Tool values only, not contractor code counts.
- Use the tool on sheet 2.a2 RD RI RT Calculation Tool in to aid in the determination of these very important reuse factors.
- Use logical code for all SLOC counts.
- Links are provided for CA rankings at the top of each column
- Use a separate line for each CSCI. If more than one language is used within the CSCI, use a different line for each language.
- See Notes at the bottom of each page for explanation of columnar headings.

Contractor WBS NO.	Item ID	CSCI Description	Logical SLOC	ITEM SIZE DATA										MONTH SDR TO CSCI TEST	SOURCE OF S/W	CONTR	PERCENT SUBCONTRACT			
				SOURCE LANGUAGE	DELIVERED NEW CODE		PRE-EXISTING CODE													
					UNIQUE SLOC	AUTO GEN SLOC	TOTAL UN-MODIFIED SLOC	TOTAL MODIFIED SLOC	TOTAL DELETED SLOC	%RD	%RI	%RT	ESLOC				1ST TIER	CONTR	2ND TIER	CONTR
			0										0							

RD/RI/RT Calculation Tool

SEER-SEM Rework Percentage Calculation

Compute redesign, reimplementation and retest percentages based on detailed rework factors.

Step 1: Set Redesign Factors

Redesign Breakdown

Formula: $A$17*A+$A$18*B+$A$19*C+($A$20*D+$A$21*E)*(1-($A$17*A+$A$18*B))

	Result Redesign Percentage		0.00%	0.00%	0.00%	
Weight	**Redesign Component**		**Least**	**Likely**	**Most**	**Percentage of the existing software that…**
0.22	Architectural Design Change	A	0%	0%	0%	... requires architectural design change
0.78	Detailed Design Change	B	0%	0%	0%	... requires detailed design change
0.5	Reverse Engineering Required	C	0%	0%	0%	... requires reverse engineering
0.225	Redocumentation Required	D	0%	0%	0%	... requires redocumentation
0.075	Revalidation Required	E	0%	0%	0%	... requires revalidation with the new design

Step 2: Set Reimplementation Factors

Reimplementation Breakdown

Formula: .37* A + .11*B +.52*C

	Result Reimplementation Percentage		0.00%	0.00%	0.00%	
Weight	**Inputs**		**Least**	**Likely**	**Most**	**Percentage of the existing software that…**
0.37	Recoding Required	A	0%	0%	0%	... requires actual code changes
0.11	Code Review Required	B	0%	0%	0%	... requires code reviews
0.52	Unit Testing Required	C	0%	0%	0%	... requires unit testing

Step 3: Set Retest Factors

Retest Breakdown

Formula: .10*A + .04*B + .13*C + .25*D + .36*E + .12*F

	Result Retest Percentage		0.00%	0.00%	0.00%	
Weight	**Inputs**		**Least**	**Likely**	**Most**	**Percentage of the existing software that…**
0.1	Test Plans Required	A	0%	0%	0%	... requires test plans to be rewritten
0.04	Test Procedures Required	B	0%	0%	0%	... requires test procedures to be identified and written
0.13	Test Reports Required	C	0%	0%	0%	... requires documented test reports
0.25	Test Drivers Required	D	0%	0%	0%	... requires test drivers and simulators to be rewritten
0.36	Integration Testing	E	0%	0%	0%	... requires integration testing
0.12	Formal Testing	F	0%	0%	0%	... requires formal demonstration testing

Tool provided in CAAG datasheet package to assist in RD/RI/RT population

Definitions

- **Average Bias:**

$$\%Bias = 100 \times \frac{1}{n} \sum \frac{y_i - \hat{y}}{\hat{y}}$$

- **Pearson R^2:** Pearson product-moment correlation squared (between actual and estimated costs), which is the percentage of variation in actual costs that is explained by the CER.

$$R^2 = \left[\frac{n \sum y_i f(x_i) - \sum y_i \sum f(x_i)}{\sqrt{n \sum y_i^2 - \left(\sum y_i\right)^2} \sqrt{n \sum f(x_i)^2 - \left(\sum f(x_i)\right)^2}} \right]^2$$

- **SPE:** Standard Percent Error. For *n* data points and *m* estimated coefficients,

$$SPE = 100 \times \sqrt{{}^{1}\!/\!{}_{(n-m)} \sum_{i=1}^{n} \left(\frac{y_i - \hat{y}}{\hat{y}} \right)^2}$$