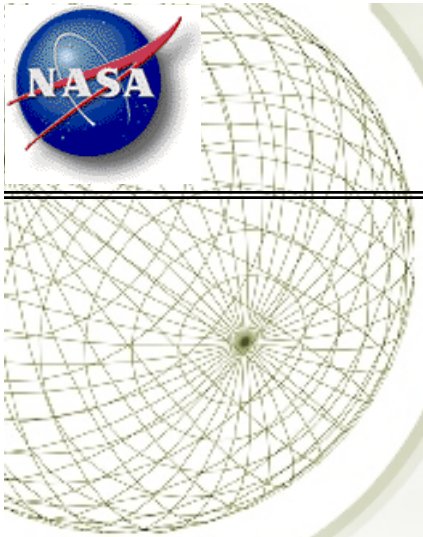


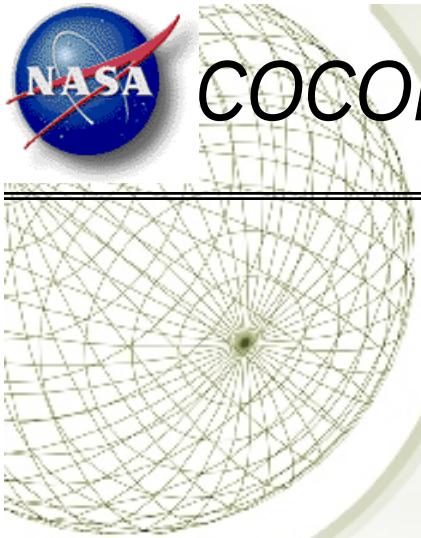
Software Development Cost Estimation

Dr. Jairus Hihn
JPL

Dr. Tim Menzies
Portland State University

Software costing is a quality issue

- Need some process V&V
  - Are their budgets ok?
  - Beware "wedge funding"
    - Thin edge of the wedge
    - Accept less money than what you need
    - Rush on,
    - Skimp on early life cycle quality work
    - Hope that you can get enough early results to secure more funding, later
- Need process V&V to find errors sooner
  - The project could find the same bug, later
  - But the older the bug, the more costly its fix
  - Lewis: IV&V costs <= 10% of software development costs
  - Therefore, need software development costs to find IV&V costs



COCOMO

- COCOMO ➡ COnstructive COst Model
- COCOMO first developed by Barry Boehm in 1981
- Effort =a * SLOC ^{b} * EM1*EM2 * EM3..
  - EM= effort multiplier: liner impact
  - A,B = tuning constants
- COCOMO II:
- Effort =a * SLOC $^{(b + SF1 + SF2 + ..)}$ * EM1*EM2 * EM3..
  - SF= scale factor; exponential impact
- This study: COCOMO-I (since COCOMO II data not public)
  - http://promise.site.uottawa.ca/SERepository/datasets-page.html

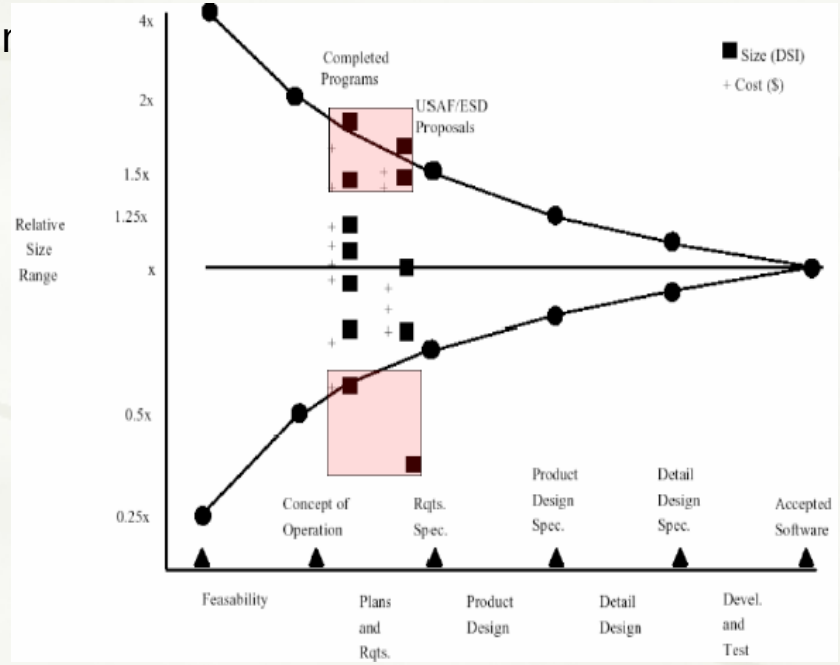
IMPORTANCE/BENEFITS

- Data at JPL indicates that
  - flight software planned effort grows by
    - 75% from Initial Confirmation
    - 55% from Confirmation Review
  - Schedule slips by 20% from Confirmation Review
  - Allocated budgets are seriously out of line with software team estimates
- The products of this research task will enable the ability to improve our performance against these metrics

Costing: an imprecise science

- Computer News:Wed June 11, 2003
- To gain control over its finances, NASA last week scuttled a new launch control system for the space shuttle.
- A recent assessment of the Checkout and Launch Control System, which the space agency originally estimated would cost $206 million to field, estimated that costs would swell to between $488 million and $533 million by the time the project was completed.

Ir



COCOMO-II

- Current high-water mark (warts and all…)
- In 15 sub-samples of 161 projects
  - PRED(30)=69% (average);
  - i.e. 69% projects estimated to within 30% of actual
- While not precise, useful for reducing variance

ACCOMPLISHMENTS

- Identified easily available datasets
- Processed and transferred contemporary flight software data to T. Menzies for analysis and model development
- Negotiated budget increase to speed up data collection from other centers
  - Identified potential data sources at GSFC and MSFC
- Verified analysis approach yields useful results
  - Completed initial analysis of 1980's NASA dataset to verify analysis approach
    1. Feature Subset Selection Can Improve Software Cost Estimation, PROMISE 05, May 15 2005, St Louis, MS.
    2. Simple Software Cost Analysis: Safe or Unsafe?, PROMISE 05, May 15 2005, St Louis, MS.
    3. Validation methods for calibrating software effort models, ICSE 2005 Proceedings, May2005, St Louis, MS.

Feature Subset Selection Can Improve Software Cost Estimation

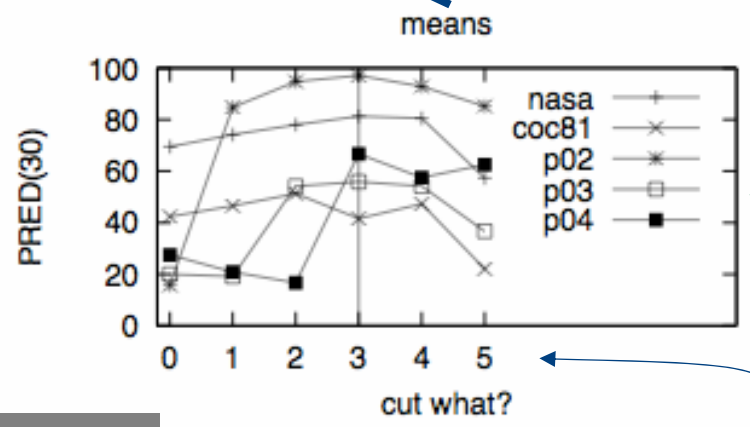
- ICSE Promise workshop, 2005
- http://timmenzies.net/pdf/promise30.pdf
- With Zhihao Chen, Dan Port, Barry Boehm

  - Standard software cost model lifecycle
    - As experience grows…
    - … and new situations encountered …
    - … add attributes to cover special situations
  - A Sisphyean task: pushing around a model of ever-increasing complexity
  - So:
    - If experience can tell you to ADD attributes
    - It should also say when to DUMP them

better predictions

better extrapolation from old to new projects

often, less variance



Increasing generality (less attributes)

a) attributes sorted by “magic” into 5 groups;
b) groups dropped one by one

Simple Software Cost Analysis: Safe or Unsafe?

- ICSE Promise workshop, 2005
- http://menzies.us/pdf/05safewhen.pdf
- With Dan Port, Zhihao Chen, Jairus Hihn

- New project cost = *delta* * (last project cost)
- *Delta* comes from COCOMO effort multipliers
  - E.g. last project:
    acap = v .high and rely=high
  - New project:
    acap = nominal, rely=low
  - New =
    old * (1/0.71 * 0.88/1.15 = 108%)
- Assumes "new" can be safely extrapolated from old
  - Is this always true?

	very low	low	nominal	high	very high	extra high
acap	1.46	1.19	1.00	0.86	0.71	
pcap	1.42.	1.17	1.00	0.86	0.70	
aexp	1.29	1.13	1.00	0.91	0.82	
modp	1.24.	1.10	1.00	0.91	0.82	
tool	1.24	1.10	1.00	0.91	0.83	
vexp	1.21	1.10	1.00	0.90		
lexp	1.14	1.07	1.00	0.95		
sced	1.23	1.08	1.00	1.04	1.10	
data		0.94	1.00	1.08	1.16	
turn		0.87	1.00	1.07	1.15	
virt		0.87	1.00	1.15	1.30	
stor			1.00	1.06	1.21	1.56
time			1.00	1.11	1.30	1.66
rely	0.75	0.88	1.00	1.15	1.40	
cplx	0.70	0.85	1.00	1.15	1.30	1.65

Extrapolation is safe only on some attributes

Sub-sampling experiments:
Learn models from N * 90% samples
Some attributes (e.g. X1) have unstable coefficients
Some attributes (e.g. X2) only used sometimes

	$\beta_0 +$	$\beta_1 X_1 +$	$\beta_2 X_2 +$	$\beta_3 X_3 +$	$\beta_4 X_4$
sub sample 1 :	$23 +$	$101 X_1 +$	$21 X_2 +$	$31 X_3 +$	$41 X_4$
sub sample 2 :	$25 +$	$11 X_1 +$		$30 X_3 +$	$42 X_4$
sub sample 3 :	$24 +$	$1 X_1 +$		$32 X_3$	

3 * 90% samples

30 * 90% samples

COCOMO 81	n	β_{mean}	β_{sd}	NASA 60	n	β_{mean}	β_{sd}	
loc	30	1.2	0.1	loc	30	1.1	0.0	✔
cplx	24	1.4	0.3	stor	22	-0.8	0.8	✔
time	16	1.9	0.4	time	29	2.5	0.8	✔
pcap	29	1.7	0.5	cplx	16	1.7	1.0	✔
acap	29	2.1	0.5	acap	28	2.7	1.0	✔
rely	30	2.0	0.6	data	26	3.1	1.2	✔
sced	25	2.9	0.7	turn	17	1.8	1.5	✘
virt	22	2.3	0.8	modp	16	-1.8	1.6	✘
vexp	24	3.0	1.2	vexp	26	-5.6	2.3	✘
				lexp	16	2.5	3.0	✘

9 attributes

10 attributes

- Only use <u>some</u> attributes can extrapolate from old to new projects
  - Many attributes missing in the sub-samples
  - Many attributes have wildly varying effects in different sub-samples

Validation methods for calibrating software effort models

- ICSE 2005

Validation methods for calibrating software effort models

- ICSE 2005
- http://menzies.us/pdf/04coconut.pdf
- With Dan Port, Zhihao Chen, Jairus Hihn Sherry Stukes

Stream of new projects

- COCONUT= COCOmo, Not Unless Tuned: a baseline calibration method

- Models a manager learning local pricing information

- Deliberately, very simple:
  - Your new method should do better than COCONUT

How long before good estimates??

COCONUT: cocomo, not unless tuned

(effort = $a*sloc^b * em_1 * em_2 *...$)?

- **Try keeping effort multipliers constant**
- **For i=1 to number of projects**
  - Train on 1 to i
  - Test on i+1 to N
- **For a train set,**
  - For all values of <a,b>
  - Find a' b' that minimizes error
- **For a different test set,**
  - Estimate using a' b'
  - Return PRED(20), PRED(20)
    - percentage of projects that estimate within 20/30% of actual
- **Repeat the above 30 times**
  - Randomizing order of projects, each time
  - Return mean and sd at each "i" value

```
function train() {
    least=10**32;
    for(a=2; a<=5; a += 0.2) {
      for(b=0.9; b<=1.2; b += 0.02) {
          close =use(a,b,pred);
          if (close < least) {
                  least=close;
                  a′=a;
                  b′=b   }}}
    return <a′,b′>}
```

exhaustive search: hard to argue that some other method might do better

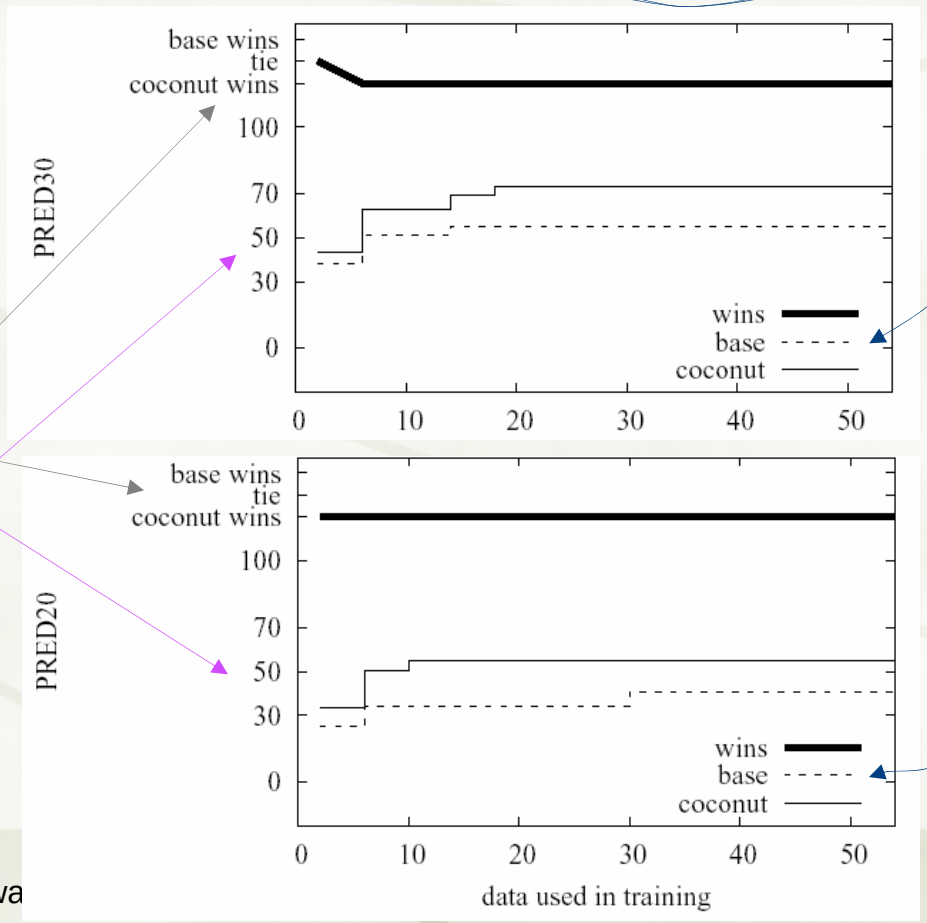
60 NASA COCOMO-I Projects in PROMISE repository

Straw man

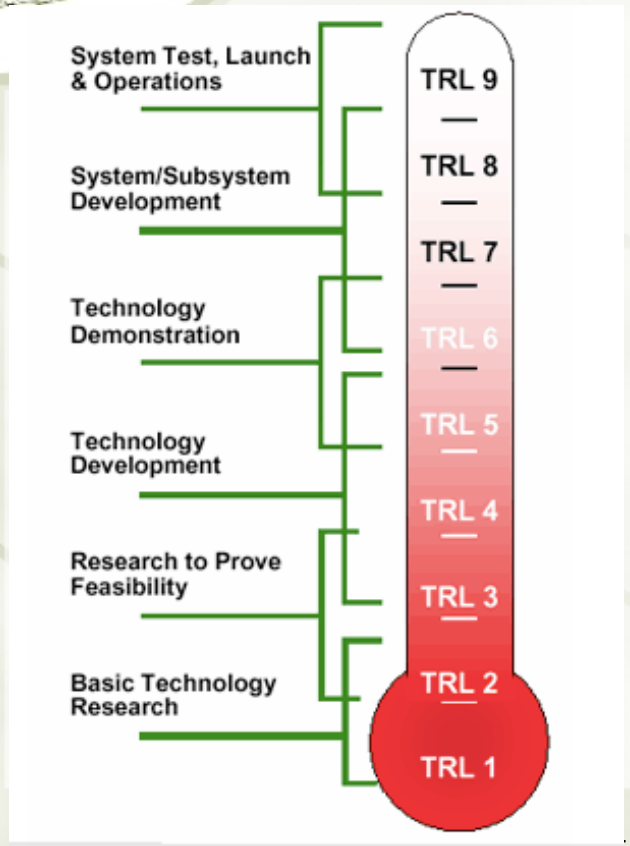
*base = a*sloc^b*

*cocomo81 = a*sloc^b * em_1 * em_2 *...*

- 30 repeats (randomizing the order)
- Use t-tests to compare
  - PRED(N) using coc81 or base
  - PRED(N) after N1 or N2 projects
- Significant changes up to
  - 18 projects for PRED(30)
  - 30 projects for PRED(20)



Technology Readiness Level of the Work



- TRL 6 or 7
- 7:
  - System prototype demonstration in a space environment
- 6:
  - System/subsystem model or prototype demonstration in a relevant environment (Ground or Space)

Potential Applications

- All NASA software

Availability of data or case studies

- PROMISE repository of software engineering data sets
  - Data sets in COCOMO-I format
- COCOMO 81:
  - http://promise.site.uottawa.ca/SERepository/datasets/cocomo81.arff
- COCOMO NASA:
  - http://promise.site.uottawa.ca/SERepository/datasets/cocomonasa_v1.arff

Barriers to research or applications

- Getting data
- Acknowledgements
  - Pat Callis, Ken McGill, Bill Jackson
  - Some recent supplemental funding for us to chase more data

NEXT STEPS

- Coordinate of IVV&V task with OCE Analogy Based Software Cost estimation
- Complete model development and analysis for Deep Space Software Cost model based on JPL data
- Finalize plans and collect available data from other NASA Centers
- Generate additional domain models as data becomes available
- Provide data to IV&V and One NASA Repositories
- Continue publishing and presenting results