Thesis Defense

Oussama El-Rawas

Software Process Control Without Calibration

Committee Members

Dr. Tim Menzies (Chair)

Dr. Hany Ammar

Dr. Bojan Cukic

West Virginia University,

Lane Departement of Computer Science and Electrical Engineering

Publications from this work

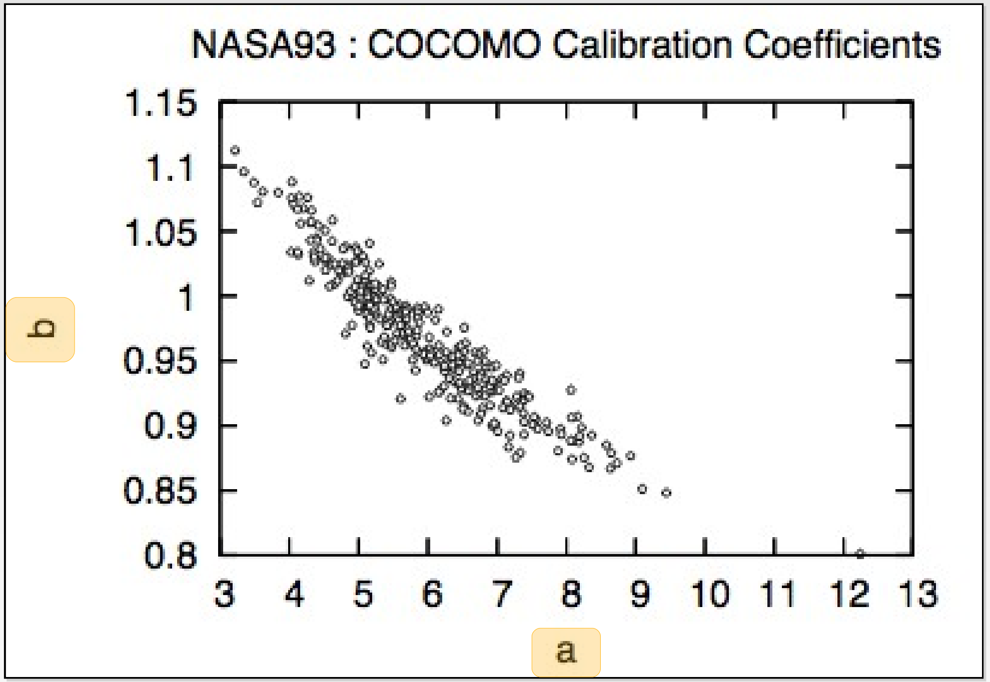
- **ICSE'09**: How to avoid drastic software process change. [Menzies, Williams, ElRawas, Boehm, Hihn 2009]
- **ASE'07**: The business case for automated software engineering. [Menzies, ElRawas, Hihn, Feather, Boehm, Madachy 2007]
- **ASE'07 workshop**: On the value of stochastic abduction, *International Workshop on Living with Uncertainty*. [Menzies, ElRawas, Baker, Hihn, Lum 2007a]
- **ICSP'08**: Accurate estimates without calibration. [Menzies, ElRawas, Boehm, Madachy, Hihn, Baker, Lum 2008]

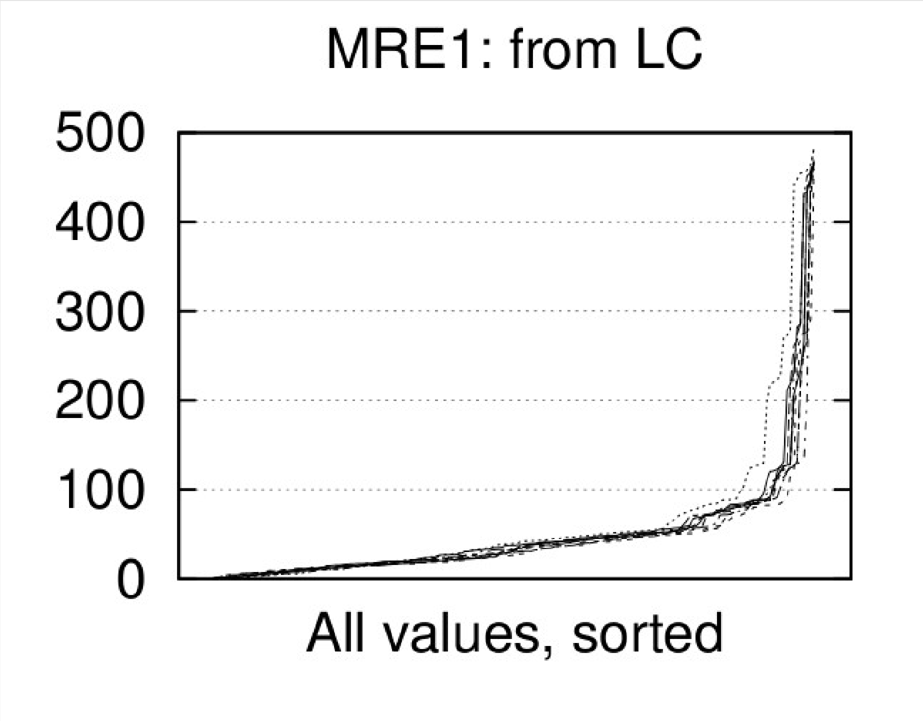
Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

Variance in Calibrations

- Much larger than reported [Baker 2007]:
  - For 93 NASA records
- Causing Estimation using Standard methods (LC) to have a high Magnitude of Relative Error (MRE = abs(predicted - actual) /actual)





$$a * \left(KSLOC^{\left(b+0.01*\Sigma_{i=1}^{5} SF_i\right)}\right) * \left(\prod_{j=1}^{17} EM_j\right)$$

Can we avoid calibration?

- Model estimates depedent on modelCalibration when…

  Estimate = projectDetails * **modelCalibration**

- But we assume that

  Estimate = **projectDetails** * modelCalibration

- Accurate Calibration vs. Data Drought: We need to avoid data dependency
- Monte Carlo [Metropolis 1953] sampling over
  - the space of possible calibrations
  - the project options
- Apply AI search to select
  - Project options that most improve the estimate
  - But do not try to control the calibrations (variation in project details assumed larger)
- Models used: COCOMO II (effort), COQUALMO (defects), Schedule and Threat models [Boehm 2000]

The Models

effort multi.
defect removal
scale factors

- COCOMO II

$$Effort = A * (c_i)(Size)^{Exponent}$$

$$PM = a * \left(KSLOC^{(b+0.01*\sum_{i=1}^{5} SF_i)}\right) * \left(\prod_{j=1}^{17} EM_j\right)$$

- COQUALMO
  (defects introduced)*(defects reduction ratio)
  {*requirements, design, coding*}

- Months (Schedule)

$$F = d + 0.2 * (0.01 * (\sum_{i=1}^{5} SF_i))$$

$$TDEV = \left[c * (PM_{NS})^F\right] * SCED\% \div 100$$

SCED	SCED%
1	75
2	85
3	100
4	130
5	160

- Threat

```
Total_threat =
(Schedule_threat + Product_threat + Personnel_threat +
 Process_threat + Platform_threat + Reuse_threat)/3.73
```

feature	low	high	feature	low	high
aa	1	6	time	3	6
peer	1	6	stor	3	6
ett	1	6	pvol	2	5
prec	1	6	acap	1	5
flex	1	6	pcap	1	5
resl	1	6	pcon	1	5
team	1	6	apex	1	5
pmat	1	6	plex	1	5
rely	1	5	ltex	1	5
cplx	1	6	tool	1	5
data	2	5	sced	1	5
docu	1	5	site	1	6
ruse	2	6	Ksloc	1	980

Default ranges for attributes in the USC models. VL = 1, XH = 6. Also used for "ALL" Project

	rely= very low	rely= low	rely= nominal	rely= high	rely= very high
sced= very low	0	0	0	1	2
sced= low	0	0	0	0	1
sced= nominal	0	0	0	0	0
sced= high	0	0	0	0	0
sced= very high	0	0	0	0	0

Sample threat table

What is the space of project options?

project	ranges feature	low	high	values feature	setting
OSP: Orbital space plane	prec	1	2	data	3
	flex	2	5	pvol	2
	resl	1	3	rely	5
	team	2	3	pcap	3
	pmat	1	4	plex	3
	stor	3	5	site	3
	ruse	2	4		
	docu	2	4		
	acap	2	3		
	pcon	2	3		
	apex	2	3		
	ltex	2	4		
	tool	2	3		
	sced	1	3		
	cplx	5	6		
	KSLOC	75	125		
JPL flight software	rely	3	5	tool	2
	data	2	3	sced	3
	cplx	3	6		
	time	3	4		
	stor	3	4		
	acap	3	5		
	apex	2	5		
	pcap	3	5		
	plex	1	4		
	ltex	1	4		
	pmat	2	3		
	KSLOC	7	418		

project	ranges feature	low	high	values feature	setting
OSP2	prec	3	5	flex	3
	pmat	4	5	resl	4
	docu	3	4	team	3
	ltex	2	5	time	3
	sced	2	4	stor	3
	KSLOC	75	125	data	4
				pvol	3
				ruse	4
				rely	5
				acap	4
				pcap	3
				pcon	3
				apex	4
				plex	4
				tool	5
				cplx	4
				site	6
JPL ground software	rely	1	4	tool	2
	data	2	3	sced	3
	cplx	1	4		
	time	3	4		
	stor	3	4		
	acap	3	5		
	apex	2	5		
	pcap	3	5		
	plex	1	4		
	ltex	1	4		
	pmat	2	3		
	KSLOC	11	392		

“Values” = fixed

“Ranges”= Loose (select within these ranges)

Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

Search-based Software Engineering

- Using search algorithms to tackle SE issues [Harman 2007, Clark 2003]
  - Project planning and cost estimation [Aguilar-Ruiz 2001, Antoniol 2005]
  - Software Testing [Briand 2005, Li 2007]
  - Automated maintenance [Mitchell 2006, O'Keeffe 2006]
- Standard search algorithms used include Genetic Algorithms [Holland 1992] and Simulated annealing [Kirkpatrick 1983]
- Other Algorithms used that go beyond Harman [Harman 2007] algorithms (e.g. Seesaw [Menzies 2009], XOMO and Tar3 [Menzies 2005])

What is STAR?

- SAMPLE: Using an AI search algorithm
- DISCRETIZE: remove continuous variables
- CLASSIFY: into 2 classes, best and rest (BORE) [Clark 2005]
- RANK: Support based bayesian ranking
- PRUNE: or back select applying policies by rank
- REPORT: returning the best set of policies

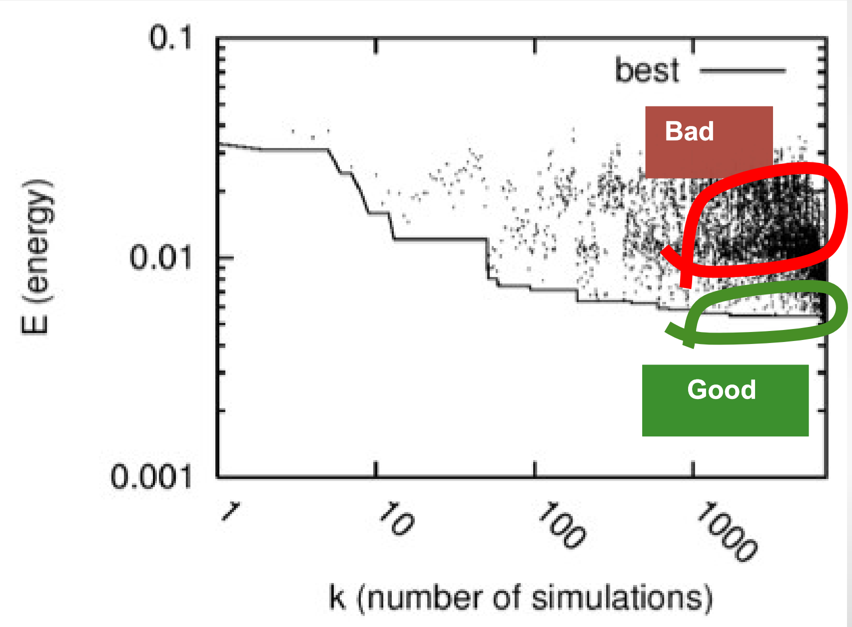
Project ranges Internal ranges

SAMPLE: Simulated Annealing [Kirkpatrick 1983]

- Best = anything
- Run from "hot" to "cool"
  - Find something in the neighborhood of best
  - If better, then new best
  - Else
    - When "hot", maybe jump to worst
    - When "cooler", don't be so stupid
- As we cool, SA converges to greedy hill-climbing
- Accumulate the total energy seen for each setting

```
function sa(kmax)

s := s0; e := E(s)             // Initial state, energy.
sb := s; eb := e               // Initial "best" solution
k := 0                         // Energy evaluation count.
while k < kmax and e > emax  // Loop
  sn := neighbour(s)           //   Pick some neighbour.
  en := E(sn)                  //   Compute its energy.
  if en < eb then              //   Is this a new best?
    sb := sn; eb := en         //     Yes, save it.
  if random() < P(e, en, temp(k/kmax))
   then s := sn; e := en       //     Maybe jump
  k := k + 1                   //   One more evaluation
    done
return sb                      // Return best
```



Ranking and the Fitness Function

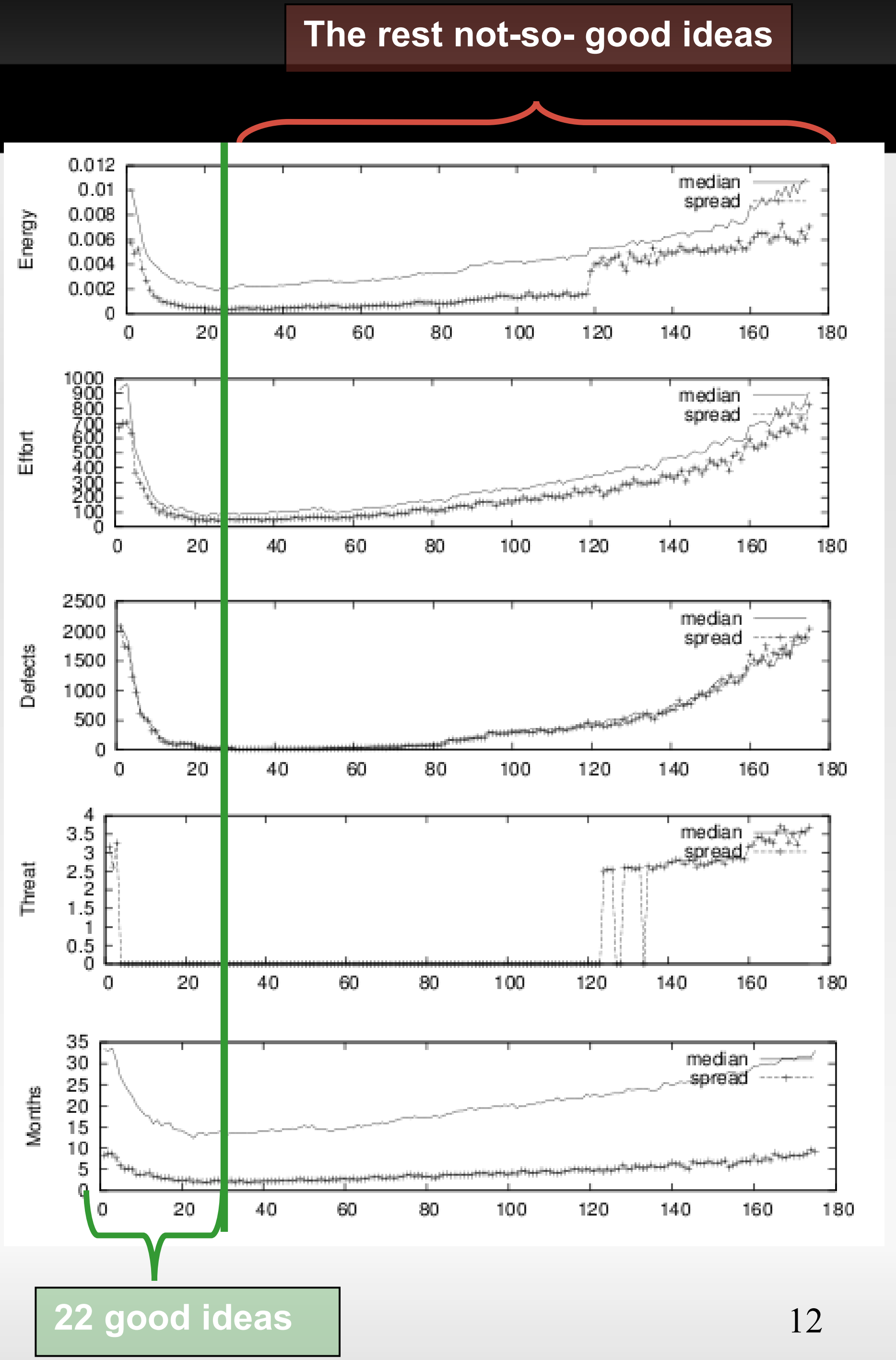
$$P(best|E) * support(best|E) = \frac{like(x|best)^2}{like(x|best) + like(x|rest)}$$

Bayes rule + support

$$E = \frac{\left(\sqrt{(\overline{Ef}*\alpha)^2 + (\overline{De}*(\beta+RD))^2 + (\overline{Th}*\gamma)^2 + (\overline{Mo}*\delta)^2}\right)}{\sqrt{\alpha^2 + (\beta+RD)^2 + \gamma^2 + \delta^2}}$$

- Median = 50% percentile
  - Spread = (75-50)% percentile
  - Small spread means stable across space of possible calibrations

- “Policy point” : smallest i with lowest E (Energy). Note that $0 \leq E \leq 1$



Analysis Types

		strategic?	tactical?
scale	prec: have we done this before?	✓	
factors	flex: development flexibility		✓
(exponentially	resl: any risk resolution activities?		✓
decrease	team: team cohesion		
effort)	pmat: process maturity	✓	
upper	acap: analyst capability	✓	
(linearly	pcap: programmer capability	✓	
decrease	pcon: programmer continuity	✓	
effort)	apex: analyst experience	✓	
	plex: programmer experience		✓
	ltex: language and tool experience		✓
	tool: tool use	✓	✓
	site: multiple site development	✓	✓
	sced: length of schedule		✓
lower	rely: required reliability		
(linearly	data: secondary memory storage requirements		
increase	cplx: program complexity	✓	✓
effort)	ruse: software reuse	✓	✓
	docu: documentation requirements	✓	✓
	time: runtime pressure		
	stor: main memory requirements		
	pvol: platform volatility		✓
hline COQUALMO	aa: automated analysis	✓	✓
defect removal	ett: execution-based testing tools	✓	✓
methods	peer: peer reviews	✓	✓

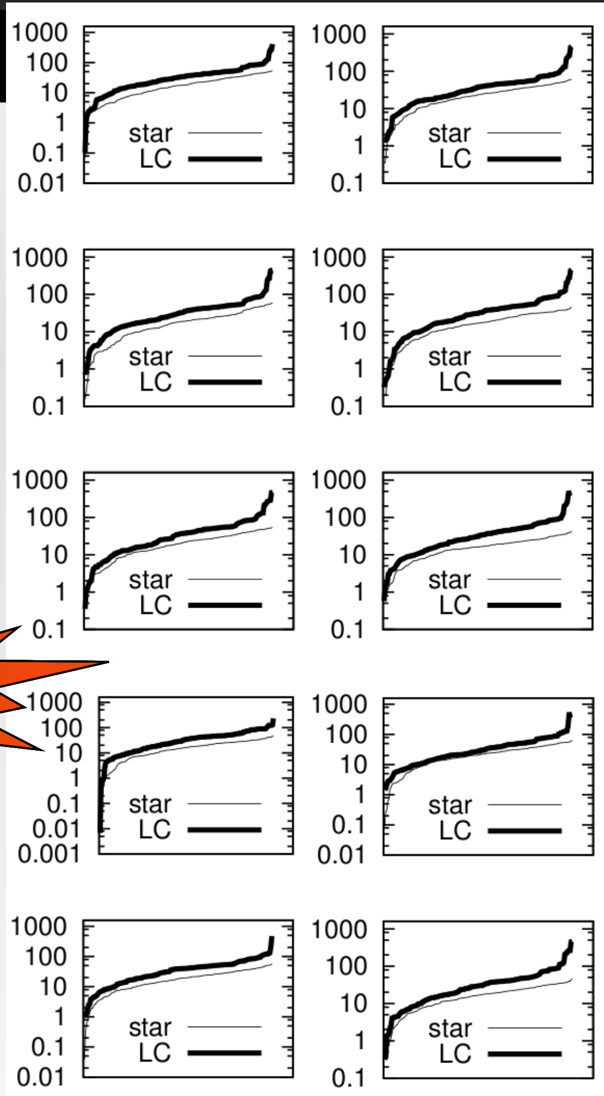
- Strategic: High level institutional change, longer term
- Tactical: within project timeline, shorter term
- All/default: Use all the attributes, its all fair game (not realistic)

Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

Sanity Check

Search vs. Standard Methods



- At the “policy point”, STAR’s random solutions are surprisingly close to LC
- LC : learn *impact[i]* via regression (JPL data)
- STAR: no tuning, randomly pick *impact[i]*
- *Median δ = (estimate(STAR) − estimate(lc))*

cast study	control method	δ %
OSP	strategic	21.78
ground	tactical	22.17
All	strategic	22.95
All	tactical	23.14
ground	strategic	23.38
flight	tactical	25
OSP2	strategic	26.6
OSP	tactical	27.71
flight	strategic	28.47
OSP2	tactical	32.07

Note: no calibration

- STAR Estimates Close to those produced by the reference models

Model	STAR Median	STAR Spread	Refrence Model	Differece	Difference%
COCOMO II	677.83	159.75	840.3	-162.47	-19.33
COQUALMO	540.39	313.22	417.25	+123.14	+29.51
Months	23.97	2.52	24.2	-0.23	-0.95
Threat	0	0	0	0	0

Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

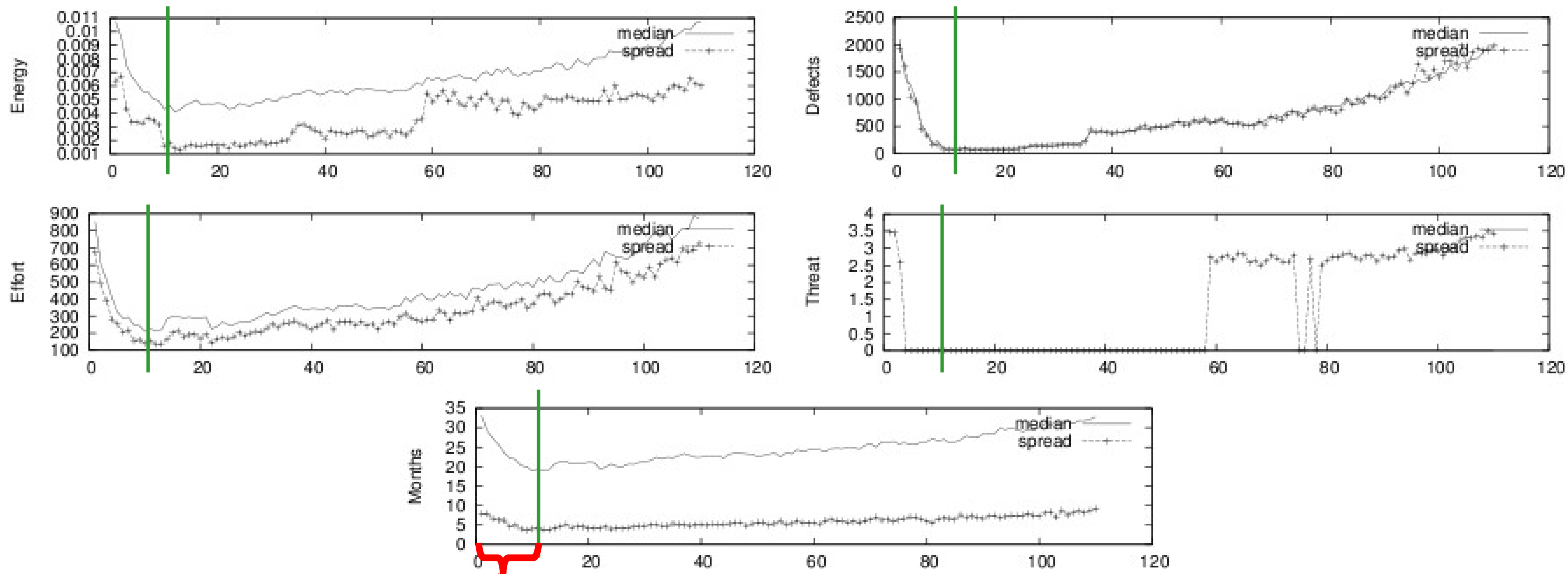
"Sociology beats Technology"?

- ICSE 2007 panel
  - Tim Lister (co-author of Peopleware" [DeMarco 1987])
- Focus less on new ASE tools and more on management / sociological factors
- E.g. More important than "software tools"
  - Any one of 1,2,3,4,5,6,7,8
  - Any two of 10,11,12,..,22
- Advanced ASE tools are aa = 5,6 and ett = 5,6. These are denoted by ▲▲

id	features	relative weight
1	Personnel/team capability	3.53
2	Product complexity	2.38
3	Time constraint	1.63
4	Required software reliability	1.54
5	Multi-site development	1.53
6	Doc. match to life cycle	1.52
7	Personnel continuity	1.51
8	Applications experience	1.51
9	Use of software tools	1.50
10	Platform volatility	1.49
11	Storage constraint	1.46
12	Process maturity	1.43
13	Language & tools experience	1.43
14	Required dev. schedule	1.43
15	Data base size	1.42
16	Platform experience	1.40
17	Arch. & risk resolution	1.39
18	Precedentedness	1.33
19	Developed for reuse	1.31
20	Team cohesion	1.29
21	Development mode	1.32
22	Development flexibility	1.26

Relative impact on development effort.
Regression analysis of 161 projects.
Boehm e.tal. 2000

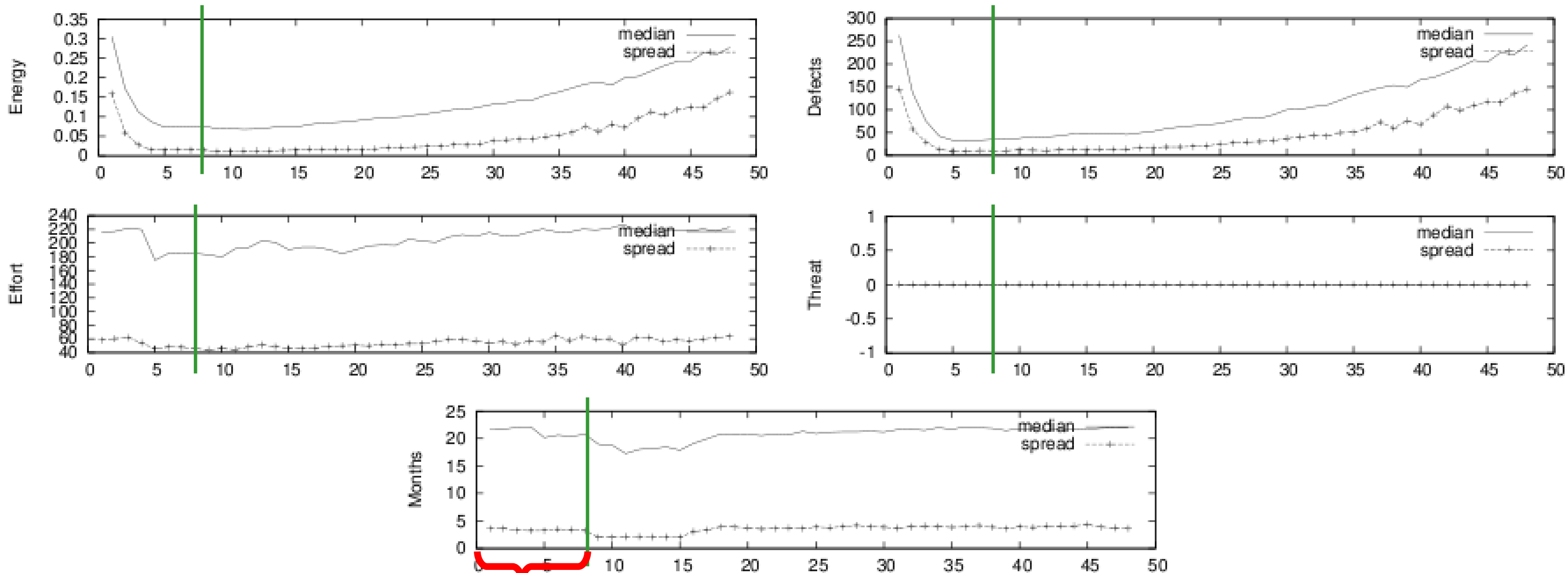
JPL Flight systems: Tactical



X	feature = range	
1	site = 6	
2	ltex = 4	
3	plex = 4	
4	execution-testing-and-tools = 6	▲▲
5	resl = 6	
6	automated-analysis = 6	▲▲
7	flex = 6	
8	peer-reviews = 6	
9	ruse = 2.5	
10	pvol = 4	
11	cplx = 3.5	

Figure 4.7: Tactical analysis: flight results.

OSP2: Tactical



X	feature = range
1	automated-analysis = 6 ▲▲
2	execution-testing-and-tools = 6 ▲▲
3	peer-reviews = 6
4	ltex = 5
5	docu = 4
6	peer-reviews = 5.5
7	docu = 3.5
8	sced = 3

Figure 4.16: Tactical analysis: OSP2 results.

ASE tools and general effects

- ASE more important for more constrained projects, but important for the vast majority.
- ASE needs to be applied with high levels, in conjunction with social factors (Technology and People work together)
- Reductions much larger for less constrained projects (exploiting uncertainty)
- Best policies a small percentage of overall space
- This result is also seen in our ASE `07 papers.

project	ALL			OSP			OSP2			flight			ground		
policies	a	s	t	a	s	t	a	s	t	a	s	t	a	s	t
models	percentage of original value														
effort	5	21	24	54	68	63	78	93	83	9	19	26	13	41	36
defects	1	21	4	10	12	12	17	15	14	3	10	6	4	26	19
threat	0	0	80	69	*108*	70	0	0	0	0	0	0	0	0	0
months	32	57	54	70	86	79	92	94	94	40	57	59	44	73	67

Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

Analysing Future SE Trends

- Studying future trends based on the COCOMO model
- ICSE '08 panel: Barry Boehm, Vic Basili, Ray Madachy, Thomas Ostrand, Deiter Romback, Rick Selby and Elaine Weyeuker
- Facilitators: Bojan Cukic, Tim Menzies, David Raffo
- Projects assumed Nasa like, Mission critical, contractor built with long lifetime (5 to 10 years)
- Trends of Attributes: Bump, Rick, Up, Down

Analysing Future SE Trends

- Trend unresolved for PREC, STOR, and TIME => 8 distinct projects
- ASE tools (technology) use a trend that is recommended for the future (more important for large)
- Effort reductions larger in Strategic
- Defect reductions larger in Tactical (use of ASE tools)

small	medium	large	
8	8	8	Flex=5
8	8	8	Resl=5
8	8	8	Ruse=2, 2.5, 3
8	8	8	Sced=1, 1.5, 2
8	7	8	Stor=3, 3.5, 4
8	8	8	Tool=5
8	8	7	Cplx=3, 3.5, 4
7	7	8	Aa=6
6	6	8	Ett=6
5	7	7	Peer=6

Figure 5.17: Tactical strategy

small	medium	large	
8	8	8	Acap=5
8	8	8	Apex=4.5
8	8	8	Pmat=4.5, 5
8	8	8	Prec=4.5, 5
8	8	8	Site=4.5
8	8	6	Pcon=3
0	3	3	Aa=6
0	2	3	Ett=6
1	1	3	Peer=6

Figure 5.16: Strategic strategy

small: KLOC = 25 to 75
medium: KLOC = 250 to 750
large: KLOC = 2500 to 7500

Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

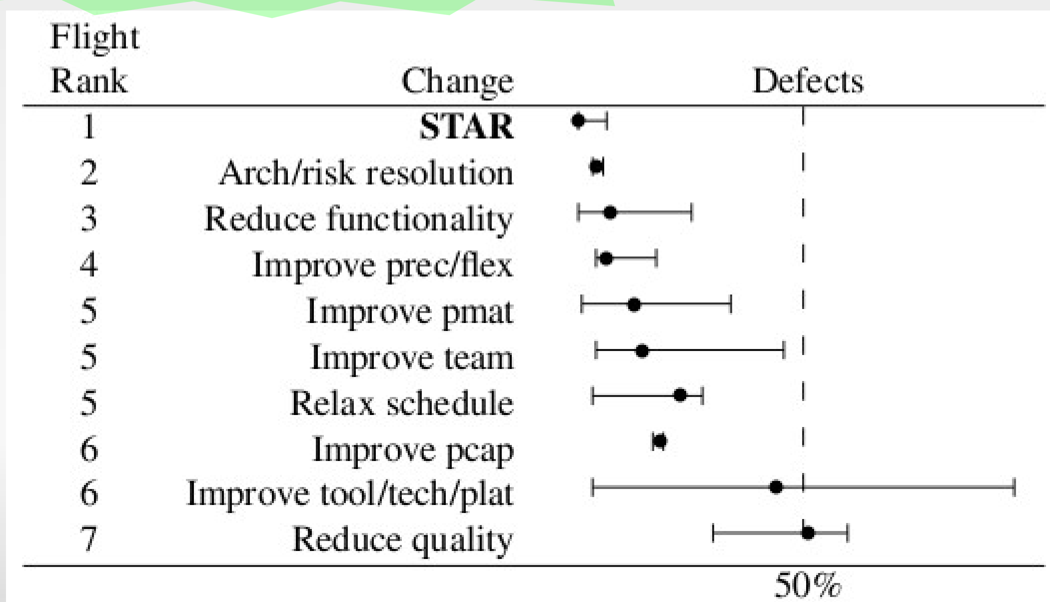
Drastic vs. Conservative control

- Conflict analysis with S-Cost [Boehm 1999]
  - Override project parameters to resolve requirement conflicts
  - Can have detremental effects on project
- Conservative change defined as change within project limits => recommendations from STAR

	Drastic change	Possible undesirable impact
1	Improve personnel	Firing and re-hiring personnel leading to wide-spread union unrest.
2	Improve tools, techniques, or development platform	Changing operating systems, IDEs, coding languages
3	Improve precedentness / development flexibility	Changing the goals of the project and the development method.
4	Increase architectural analysis / risk resolution	Far more elaborate early life cycle analysis.
5	Relax schedule	Delivering the system later.
6	Improve process maturity	May be expensive in the short term.
7	Reduce functionality	Delivering less than expected.
8	Improve the team	Requires effort on team building.
9	Reduce quality	Less user approval, smaller market.

Drastic vs. Conservative control

- Conservative change is better overall
  - ranking higher than drastic in most cases
- Drastic Change does better in some cases
  - Bad side effects (Reduce quality: quick but problematic)
- Drastic change not that great



Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

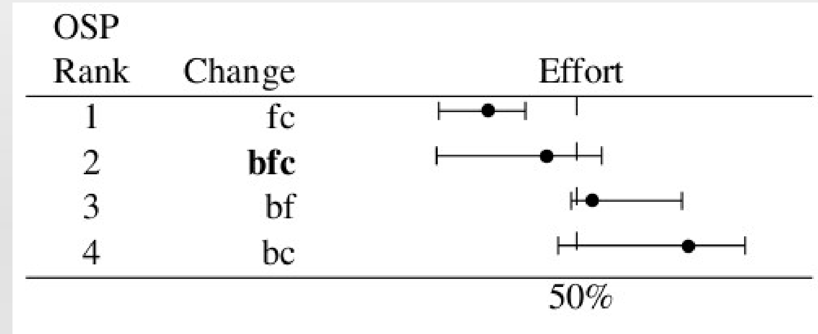
Better, Faster, Cheaper

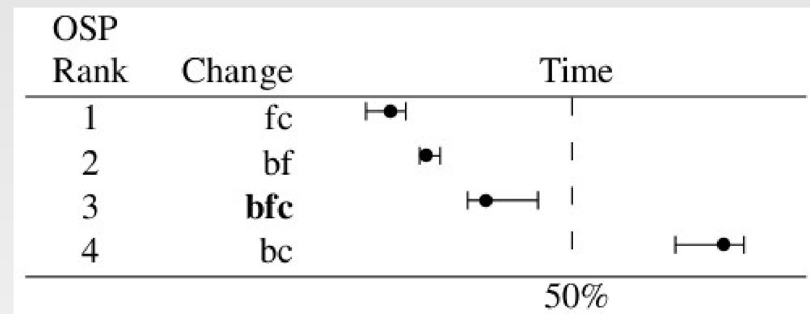
- Pushed by Dan Goldin at NASA
  - All three can be achieved
  - Implementation of BFC was closer to FC [IFPTE 2003]
- Conventionally, it's more like Better, Faster, Cheaper: pick any two (ie. BF, FC or BC)
- Applied by changing the weighting of models

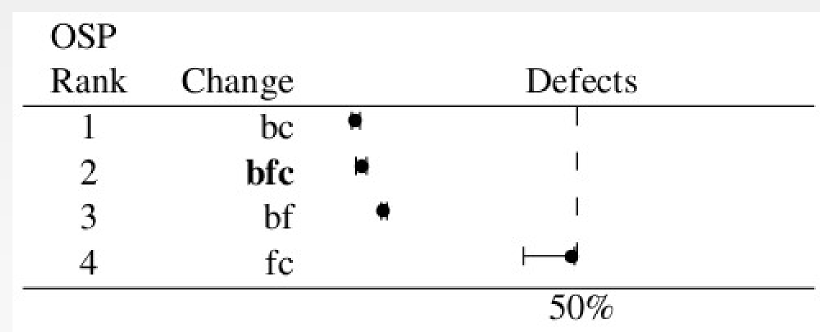
$$E = \frac{\left(\sqrt{(\overline{Ef}*\alpha)^2+(\overline{De}*(\beta+RD))^2+(\overline{Th}*\gamma)^2+(\overline{Mo}*\delta)^2}\right)}{\sqrt{\alpha^2+(\beta+RD)^2+\gamma^2+\delta^2}}$$

Better, Faster, Cheaper

- BFC ”jack of all trades, master of none”, i.e. trying to fix everything yeilds compromises
- FC bad for defects, but pushes projects out quicker (especially for more constrained)
- BC Drives defects down, but development time is worst







Road map

- Motivation
- Method
- Sanity Check
- Case Studies
  - NASA case study and ASE tools
  - Future Trends
  - Drastic vs Conservative Change
  - Better, Faster, Cheaper
- Conclusions and Future work

Conclusions

- No calibration data = Not a road block
  - If data present, use it
  - Else can still work with space of tunings
  - Uncertainty exploited to our advantage
- Search based tools can evaluate practices
  - Technology vs. People: work together
  - Technology use is recommended for the future
  - Drastic Change not that great
  - Trying to fix everything yeilds compromises

Future work

- Method being further developed
  - different search engines [Menzies 2009, Green 2009]
  - different fitness functions [Green 2009]
  - to evaluate different practices [Orrego 2009]
- Fix/Remove Threat Model

References

- [Aguilar-Ruiz 2001] : J. Aguilar-Ruiz, I. Ramos, J.C. Riquelme, and M. Toro. An evolutionary approach to estimating software development projects. Information and Software Technology, 43(14):875–882, December 2001.
- [Antoniol 2005] : G. Antoniol, M. Di Penta, and M. Harman. Search-based techniques applied to optimization of project planning for a massive maintenance project, Proceedings of the 21st IEEE International Conference onSoftware Maintenance, pages 240–249, Sept 2005.
- [Baker 2007] : Dan Baker. A hybrid approach to expert and model-based effort estimation. Master's thesis, Lane Department of Computer Science and Electrical Engineering, West Virginia University, 2007.
- [Boehm 1999] :] Barry Boehm and Hoh In, Conflict analysis and negotiation aids for cost-quality requirements, 1999.
- [Boehm 2000] : Barry Boehm, Ellis Horowitz, Ray Madachy, Donald Reifer, Bradford K. Clark, Bert Steece, A. Winsor Brown, Sunita Chulani, and Chris Abts. Software Cost Estimation with Cocomo II. Prentice Hall, 2000.

References

- [Briand 2005] : Lionel C. Briand, Yvan Labiche, and Marwa Shousha. Stress testing real-time systems with genetic algorithms. In GECCO ’05: Proceedings of the 2005 conference on Genetic and evolutionary computation, pages 1021–1028, New York, NY, USA, 2005. ACM.
- [Clark 2003] : J. Clark, J. J. Dolado, M. Harman, R. M. Hierons, B. Jones, M. Lumkin, B. Mitchell, S. Mancoridis, K. Rees, M. Roper, and M. Shepperd. Reformulating software engineering as a search problem. IEE Proceedings on Software, 150(3):161–175, 2003.
- [Clark 2005] : R. Clark. Faster treatment learning, Computer Science, Portland State University. Master’s thesis, 2005.
- [DeMarco 1987] : Tom DeMarco and Timothy Lister. Peopleware: productive projects and teams. Dorset House Publishing Co., Inc., New York, NY, USA, 1987.
- [Green 2009] : Phillip Green. Impact of value-based software engineering on software process control. Master’s thesis, WVU LCSEE dept., 2009. In progress.

References

- [Harman 2007] :   Mark Harman. The current state and future of search based software engineering, 2007 Future of Software Engineering, Pages 342–357, 2007.
- [Holland 1992] : John H. Holland. Adaptation in natural and artificial systems. MIT Press, Cambridge, MA, USA, 1992.
- [Huang 2006] : LiGuo Huang. Software Quality Analysis: A Value-Based Approach, PhD Thesis, 2006.
- [IFPTE 2003] : Ifpte report on the effectiveness of nasa's workforce & contractor policies, March 2003.
- [Kirkpatrick 1983] : S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. Science, Number 4598, 13 May 1983, 220, 4598:671–680, 1983.
- [Li 2007] : J. Li and G. Ruhe. Decision support analysis for software effort estimation by analogy. In Proceedings, PROMISE'07 workshop on Repeatable Experiments in Software Engineering, 2007.

References

- [Menzies 2005] : T. Menzies and J. Richardson. Xomo: Understanding development options for autonomy. In COCOMO forum, 2005.
- [Menzies et al. 2007] : T. Menzies, O. Elrawas, J. Hihn, M. Feather, B. Boehm, and R. Madachy. The business case for automated software engineerng. In ASE '07: IEEE/ACM international conference on Automated software engineering, pages 303–312, New York, NY, USA, 2007. ACM.
- [Menzies et al. 2007a] : T. Menzies, O. Elrawas, D. Baker, J. Hihn, and K. Lum. On the value of stochastic abduction (if you fix everything, you lose fixes for everything else). In International Workshop on Living with Uncertainty (an ASE'07 co-located event), 2007.
- [Menzies et al. 2008] : T. Menzies, O. Elrawas, B. Boehm, R. Madachy, J. Hihn, D. Baker, and K. Lum. Accurate estimates without calibration. In International Conference on Software Process, 2008.
- [Menzies et al. 2009] : Tim Menzies, Steve Williams, Oussama ElRawas, Barry Boehm, and Jairus Hihn. How to avoid drastic software process change (using stochastic stability). In International Conference on Software Engineering, 2009. To be published.

References

- [Metropolis 1953] : N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations by fast computing machines. J. Chem. Phys, 21:1087–1092, 1953.
- [Mitchell 2006] : Brian S. Mitchell and Spiros Mancoridis. On the automatic modularization of software systems using the bunch tool. IEEE Transactions on Software Engineering, 32(3):193–208, 2006.
- [O'Keeffe 2006] : Mark O'Keeffe and Mel O'Cinneide. Search-based software maintenance. European Conference on Software Maintenance and Reengineering, 0:249–260, 2006.
- [Orrego 2009] : Andres Orrego. The value of reuse in software process control. suggested paper for ICSP'09.

Live Demo

Acknowledgements

- Committee members
- Dan Baker, Omid Jalali
- Phillip Green, Steve Williams
- ICSE 2008 panel members and facilitators
- Jarius Hihn, Julian Richardson
- Parents, Friends and Family

Thank you

Questions?