Amplified Lessons from the Ant Hill –
What Ants and Software Engineers Have in Common

International Function Point Users Group

September, 2003

Joseph R. Schofield

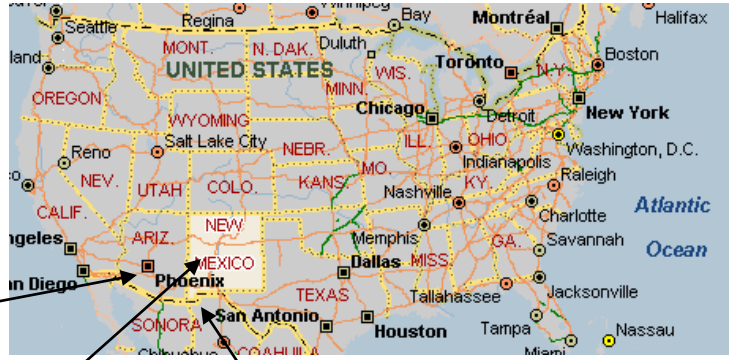
Sandia National Laboratories

Albuquerque, N. M. 87185

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy under contract DE-AC04-94AL85000.

WELCOME TO NEW MEXICO

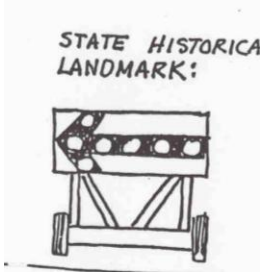
THE STATE WHERE EVERY HIGHWAY EVENTUALLY NARROWS TO A SINGLE LANE

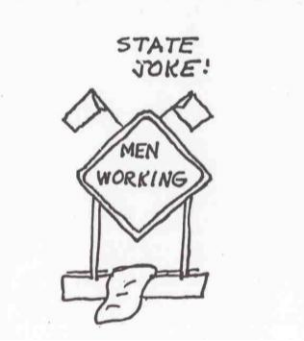


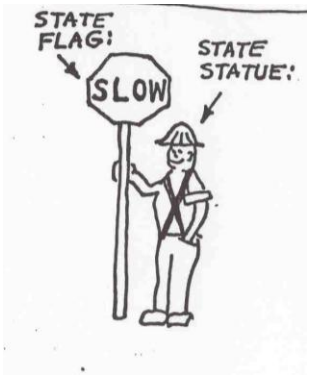
You are here!

New Mexico

Mexico





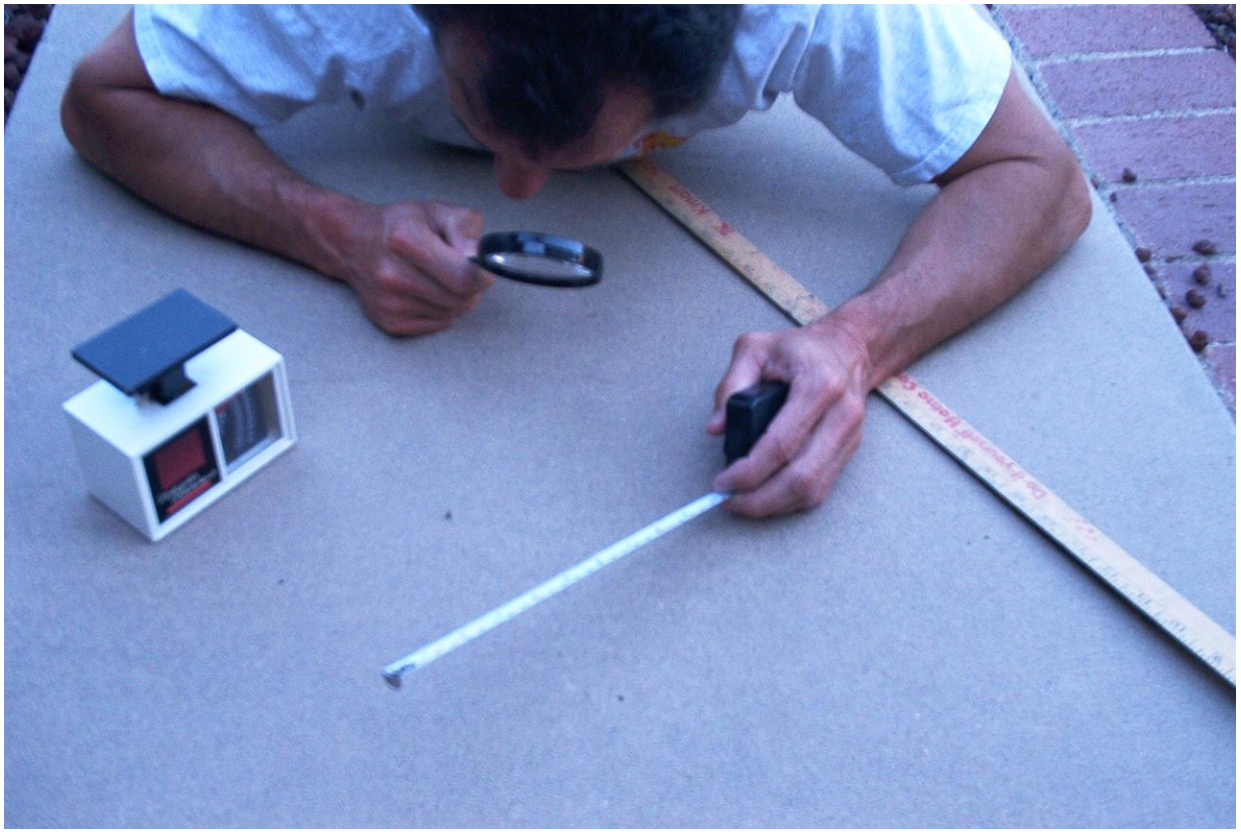


Albuquerque and Balloons: Ants and Software Engineers –

Ants and Software Engineers – Of course I'm serious!

"What are you doing?' the scientists and engineers smirked.

"What's it look like?" I retorted, "we're taking pictures of ants for an article on software productivity."



Actual data from three languages

Course	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
1	Attendee 1	89	34	67	40	102	235	23	38	168
1	Attendee 3	82	23	33	48	61	34	33	27	52
1	Attendee 4	177	119	67	85	136	276	165	112	233
1	Attendee 5	76	48	305	244	61	121	66	77	127
1	Attendee 7	46	33	17	37	60	95	129	46	186
3	Attendee 5	22	40	100	58	68	131	58	58	102
3	Attendee 6	46	20	30	42	73	82	51	72	82
2	Attendee 7	95	155	147	94	54	191	174	102	218
	Min	22	20	17	37	54	34	23	27	52
	Max	177	155	305	244	136	276	174	112	233
	% Variation	805	775	1794	659	252	812	757	415	448
	Mean	79	59	96	81	77	146	87	67	146
	Std. Dev.	47	50	95	69	28	82	60	30	66

Course	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
2	Attendee 5	193	137	48	102	107	101	116	108	73
1	Attendee 2	77	163	168	123	134	221	311	271	283
3	Attendee 1	73	37	36	95	101	631	1037	1807	388
3	Attendee 2	74	97	143	153	279	113	121	89	141
3	Attendee 4	114	71	108	80	219	56	81	73	65
	Min	73	37	36	80	101	138	51	66	134
	Max	193	163	168	153	279	207	238	178	305
	% Variation	264	441	467	191	276	150	467	270	228
	Mean	106	101	101	111	168	169	145	97	184
	Std. Dev.	51	50	58	28	78	29	69	47	71

P4	P5	P6	P7	P8	P9
227	186	306	155	61	283
13	110	63	113	84	85
34	53	51	54	61	125
61	134	99	43	58	126
289	142	122	190	383	219
150	202	185	155	118	144
30	61	116	69	43	147
197	64	158	85	87	126
169	56	23	99	73	83
49	66	103	71	51	73
13	53	23	43	43	73
289	202	306	190	383	283
2223	381	1330	442	891	388
122	107	123	103	102	141
97	56	81	49	101	65

Don't blame the *weakest link* – a look inside one classes' values

This attendee had 4 of the "largest programs"

Attendee 3 also had the shortest 7th program!

	*P1	P2	P3	P4	P5	P6	P7	P8	P9
Attendee 1	33	40	30	108	65	176	79	107	284
Attendee 2	51	52	24	72	109	166	87	145	270
Attendee 3	76	56	30	115	175	158	27	104	128
Attendee 4	60	32	31	108	94	155	72	94	235
Attendee 5	22	51	25	50	75	105	47	21	102
Attendee 6	65	27	80	45	95	141	91	60	209
Attendee 7	22	51	25	50	75	105	47	21	102
Attendee 8	65	27	80	45	95	141	91	60	209
Min	22	27	24	45	65	105	27	21	102
Max	76	56	80	115	175	176	91	145	284
% Variation	345	207	333	256	269	168	337	690	278
Mean	49	45	41	74	98	143	68	77	192
Std. Dev.	21	12	24	31	34	26	24	44	73

Four other attendees had the "largest program" in other assignments

Five other attendees had a shortest program!

Perhaps most surprising, five attendees had both a longest and shortest program in their set of assignments.

Unadjusted Function Points for the nine programs

Program	External Inputs ILFs/DETs (UFPs)	External Outputs ILFs/sDETs (UFPs)	Internal Logical Files DETs (UFPs)	Total Unadjusted Function Points	Comments
1	1/3 (3) 2/5 (4) 2/5 (4)	1/2 (4)	*1/1 (7) *1/2 (7) 1/3 (7)	36	*since the file structure was a requirement of the customer, the top level and indexing levels are counted as distinct ILFs
2	1/1 (3)	1/1 (4)		7	This program counts the lines of (ignoring blank lines and comments) and returns the number of lines of code.
3	1/1 (3)	1/5 (4)		7	List the source files for a program
4	1/2 (3)	1/5 (4)	*1/3 (7)	14	Calculate linear regression parameters *changed the data structure
5	1/1 (3)	1/1 (4)		7	Simpson's rule
6	1/4 (3) 1/5 (3)	1/4 (4) 1/4 (4) 1/4 (4)		18	Generate prediction intervals
7	1/2 (3)	1/7 (4) 1/2 (4) 1/1 (4)		15	Describe statistical relationship between two sets of numbers
8	1/2 (3)	1/2 (4)		7	Sort list by both sets of values
9	1/2 (3)	1/4 (4) 1/2 (4) 1/1 (4)		15	

A quick look at five classes wherein all attendees coded in C

Class	# of attendees	# of attendees with largest program (at least once in a set of nine)	# of attendees with smallest program (at least once in a set of nine)	# of attendees with a smallest & largest program
1	10	3	6	0
2	8	5	4	2
3	13	7	6	2
4	8	5	6	5
5	10	5	4	1

Reviewing the counters and counts, discounts the "counting" approach as a source of variation

Counting Method	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
1	Attendee 1	91	123	45	121	101	403	553	211	516
1	Attendee 2	74	97	218	194	279	406	311	181	368
1	Attendee 4	108	95	205	162	300	484	499	143	683
1	Attendee 5	193	137	182	239	127	517	353	112	510

Attendee 1's counts of 4 attendees products

Counting Method	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
2	Attendee 1	93	133	51	123	107	441	580	213	580
2	Attendee 2	74	97	218	194	279	406	310	181	368
2	Attendee 4	110	98	218	317	219	513	523	148	706
2	Attendee 5	256	172	229	310	170	675	445	122	649

Attendee 2's counts of 4 attendees products

Only attendee 2's code counter was peculiar to only attendee 5's code; **BUT, in every case, each counter counted their own code consistent with other's counting approach.**

Counting Method	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
3	Attendee 1	91	123	45	119	101	380	516	202	479
3	Attendee 2	74	96	217	194	279	406	310	181	368
3	Attendee 4	114	78	187	149	303	440	462	130	619
3	Attendee 5	193	137	181	239	127	517	353	112	510

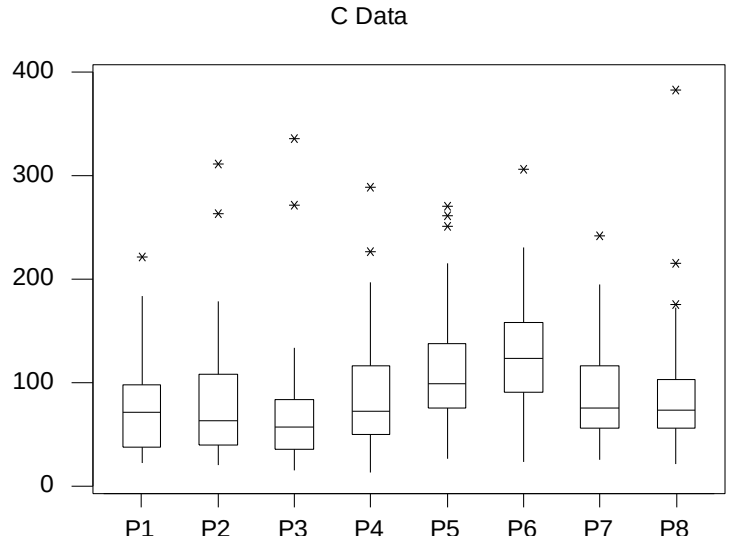
Attendee 3's counts of 4 attendees products

Counting Method	Attendee	*P1	P2	P3	P4	P5	P6	P7	P8	P9
4	Attendee 1	91	124	45	120	101	399	548	210	511
4	Attendee 2	75	98	221	197	282	408	312	182	375
4	Attendee 4	109	92	202	160	295	476	492	141	672
4	Attendee 5	193	137	182	239	127	517	353	112	510

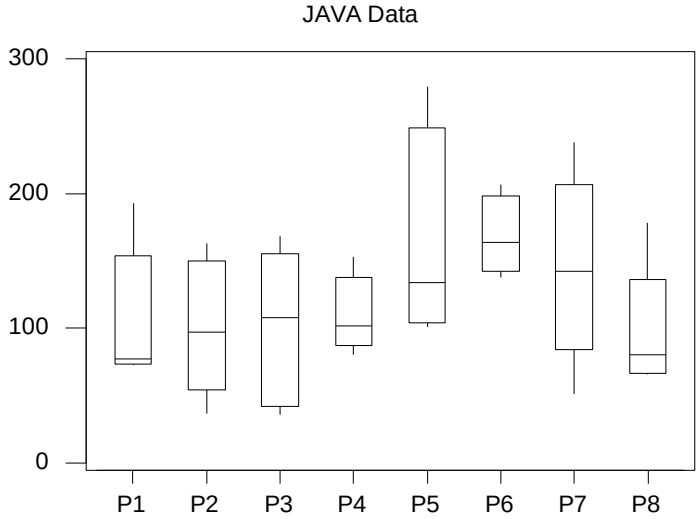
Attendee 4's counts of 4 attendees products

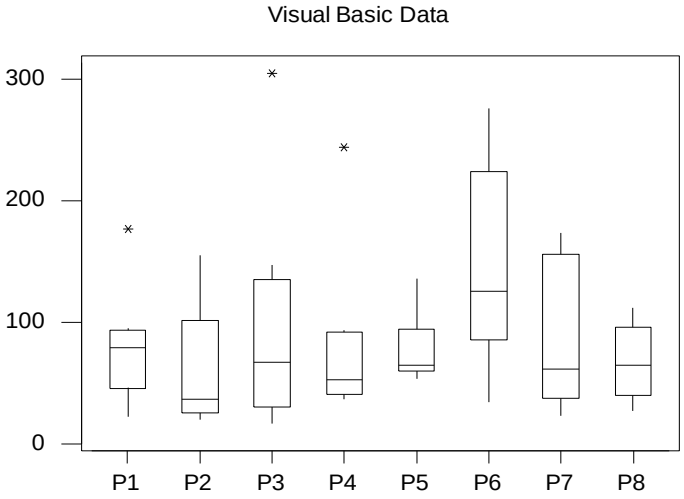
The boxplots by program & language

(note scale on C data is different than other two)



The Java and VB programs were written by software engineers with a minimal of a bachelors degree, with about ½ with advanced degrees.





And if the data isn't convincing . . .

Consider the ways of the ant . . .

Date	# of ants
6/14	5
6/17	0
6/18	4
6/19	0
6/21	1
7/9	7
7/10	3
7/12	4
Mean	3.0
Std Dev	2.5
Coe Var	.83

Activity ≠ Progress

Observation: There's often a lot of movement; but it's harder to see progress.

Lesson: Busy people doesn't mean progress. One project spent 80 percent of their time on non-scheduled tasks in the first month of their project. They knew this because they were measuring using principles from the Team Software Process^{SM} (TSP).

Reference: *Introduction to the Team Software Process*^{SM}, Watts Humphrey, 2000

Haw in *Who Moved My Cheese?*, Spencer Johnson, 1998

“Stuff” happens, resources change

Observation: Ants live 45 – 60 days – some projects live this long; others should last not longer than this.

Lesson: Establish project measures and metrics that support business intent and objectives. Establish a Project Management Office (PMO) to leverage and create consistency in project management practices.

Reference: *IT Measurement – Practical Advice from the Experts*; IFPUG; 2002

Practical Software Management; Layman, et al; 2002

CIO; Ware, et al; July 1, 2003; pgs. 22, 60, and 82

If the hole fits . . .

Observation: Some ant holes are bigger than others.

Lesson: Some software projects are bigger than others. Bigger holes accommodate more ants, and bigger projects seem to accumulate more staff, needed or not. Large projects often assume inordinate overhead.

Reference: (Michael) Hammer refers to this as non-value-adding work, and sometimes as waste or *muda*!"

Beyond Reengineering, Hammer, 1996

It's getting deep

Observation: Some ants get deep into the ground. The Messor Acculatus builds a nest as deep as 12 feet.

Lesson: Project teams often get into deep problems with performance, schedule, and cost. Many of these relate to scope / requirements (omission, misinterpretation, poorly defined), risks (not accounted for, no mitigation strategy), and skills (wrong skills, poor estimating skills, failure to include overhead).

Reference: Kerzner identifies 10 estimating pitfalls and an additional 15 risks in *Project Management*, Harold Kerzner, 1994

Getting from here to there

Observation: Ants are capable of finding the shortest path from their nest to a supply of food. They can change their path reliably if the scent on their path is disrupted or blocked.

Lesson: Most project teams are capable of finding the longest path from requirements to implementation which is why 3 out of 10 major IT projects fail. They can change their path reliably from bad to worse once they begin to fall behind.

Reference: Beckers, Deneubourg, and Goss: 1992

IT Doesn't Always Make the Grade; Informationweek; 6/16/2003; pg 16

Technically speaking . . .

Observation: Ants are classified by biologists as a type of wasp (hymenoptera formicidae). Humans are classified as people (homo sapiens). The Slave-Maker ant (polyergus rufescens) steals pupae from other ant nests to get workers.

Lesson: Software engineers are abducted from other projects; the *official* designation for this management practice is *matrixing* or *leveraging*. Either technique requires careful management of dual reporting.

Reference: Project Management Book of Knowledge; Project Management Institute, Section 9.3

Scout or Rebel?

Observation: Some ants wander "out of the box." These ants are referred to as scouts.

Lesson: "Out of the box" software engineers are called technologists, experts, pioneers, researchers, and sometimes heroes. Often they are the source of creeping requirements and technology churn.

Reference: Brooks call this growth "frills" and "embellishments."

The Mythical Man-Month, Chapter 4, Brooks

the box (in yellow)

Random processes must give way to lean defined processes

Observation: Some ants spread out in random fashion.

Lesson: Random fashion, random work, random results—this sequence requires abundant good fortune for sustained success. Consider "lean" techniques to our software processes. **Measures** can help prove the value!

Reference: Pratt & Whitney – the company joke was that the average part traveled farther inside Pratt's plants during production than it did while in airline service. The distance traveled by a part within Pratt plants (not counting distance between plants) was 18 miles; a year later they were able to reduce that to nine miles. A three minute grinding operation was accompanied by ten days of batch and queue time.

Lean Thinking, Womack & Jones, 1996

“Stuff” happens, workers need *breadcrumbs* too

Observation: Ants leave scents by which they travel; they may also spray formic acid in self-defense.

Lesson: Software teams need to leave lessons learned, artifacts, documentation, and templates for those who follow them.

Reference: CMMI®, Process Areas: Risk Management, Process and Project Quality Assurance, Decision Analysis and Resolution, Causal Analysis and Resolution, Organizational Process Performance

“Stuff” happens, maturity sets in

Observation: Ants have four stages of life: egg, larva, pupa, adult. Technologies have four stages of application: learning, applying, mastering, and maturing.

Lesson: Technologies employed by software teams also go through stages, stages of acceptance. Using risk analysis and management, determine at which “stage” you’re capable of managing technology. Can you manage the “bleeding edge” or are you better applying mature and proven technologies?

Reference: *The Next Silver Bullet*, Schofield, System Development, August 1995

“Stuff” happens, smell around

Observation: Ants can tell if other ants are from their own nest by their smell! (They don’t take kindly to “different” aromas!)

Lesson: Team members need to be close enough and interactive enough to smell each other. Depending on the smell, they may be more likely to accept and build upon the ideas of others.

“*Good ideas drive out bad ideas. The problem with most companies is that they have no good ideas.*” Bill O’Brien, former CEO Hanover Insurance

Reference: *The Fifth Discipline Fieldbook;* Senge, et al; 1994

“Stuff” happens, workers churn

Observation: Ants fight to retain control of food sources.

Lesson: Projects fight to retain resources and avoid churn. Physical separation and fragmentation may occur despite efforts to avoid *teamicide*. DeMarco & Lister identify seven potent team killers.

Reference: *Peopleware*, DeMarco & Lister, 1987, pg. 132

Software engineers are smarter than ants, right?

Observation: When ants underestimate the size of a job, they compensate with waves of more ants. Most software projects cannot afford this tactic.

Lesson: Use reliable sizing measures like Function Points to assess progress. Avoid the practice of counting lines of code as a measure of size or progress.

Reference: *A Practical, Statistical, and Criminal Look at the Use of Lines of Code as a Software Sizing Measure*, Schofield, Structured Development Forum, March, 2003

Software engineers are more resourceful than ants, right?

Observation: Ants have about 250K brain cells; humans about 10 billion. Why do they team up so well when dealing with a problem bigger than any one of them can solve?

Lesson: Getting and giving help are what peer reviews are all about. More formally performed inspections can remove up to 88 percent of defects (compare this with 30 percent through testing).

These red army ants team up to go places as a team that no individual contributor could go alone!

Reference: *Defect Management through the PSP*[SM]; Hirmanpour & Schofield; Crosstalk; September 2003

Software Quality, Jones, 1997

Software engineers can do more than ants, right?

Observation: Depending on the source, ants can carry somewhere between 20 and 100 times their own weight.

Lesson: When you find a software engineer that can carry many times her / his own weight, take really good care of ‘em!

One database of 8000 software projects showed a productivity range between 0.13 and 140 function points per person month.

Reference: *Estimating Software Costs*, Capers Jones, 1998, pg. 394

Are you mentoring?

Observation: Sometimes ants "get together" one-on-one.

Lesson: Software team members ought to get together for knowledge sharing and lessons learned to increase their knowledge. Sometimes when we're spinning our wheels instead of asking questions, we try spinning harder and faster.

Reference:

"*If something isn't working, do more of it.*" DeMarco's First Law of Bad Management, *Slack*, DeMarco, 2002

Did we really want that tool?

Observation: Ants often discard items they collect. Sometimes the seeds they obtain one day, they dispose of the next.

Lesson: Project teams often discard technologies and tools faster than ants do seeds. **Beware of what you depict as legacy since the code you develop today will be tomorrow's legacy!**

Reference: *Software Cost Estimation with COCOMO II, Barry Boehm*, 2000, Chapter 2

“Stuff” happens, get the right leader!

Observation: A single male ant plays a prominent role in the formation of the nest. And of all the female ants, only one becomes queen. After mating she loses her wings and is given the royal treatment by the remaining ants.

Lesson: Particularly on small teams, team members fill prominent roles. All roles contribute; however, someone, assigned or selected, will serve to lead the team. Train-up leaders before you need them and you’ll never be in need of them.

Reference: “An army of deer led by a lion will defeat an army of lions led by a dear.” Grant Hill from his father.

Leading with the Heart, Mike Krzyzewski, 2001

Ants – we are family . . .

Observation: All the ants in a nest are brothers and sisters—they all come from one momma!

Lesson: It may do a software team good to treat their fellow members as family (then again, maybe not)! Down time, social time, spaghetti dinners, ice cream socials, and lunches together can boost productivity in the long run.

Reference: "Can't we all just get along?" - Rodney King

You didn’t want to hear this

Observation: Ants have meetings. This fact is the worst news I uncovered during my ant research. When one ant wants to communicate with another, it merely taps the other ant on its head with its antennae.

Lesson: Avoid both these practices: agendaless meetings, and tapping other team members *upside* their heads.

Reference: Do we really need a reference?

*Anth*ology – *Anth*em or *Anth*rax?

(Up the *ante*!)

Meaningless meeting are miserable minutes

Measures provide insight

Everyone a leader "in training"

Some days you gotta fight

Keep "scouts" in check

DEFINED RANDOM

Don't let stealing abduct your project

Movement ain't progress!

Grow your family

Find the straight path, or the least crooked path

Leave scents, make sense?

Buy less, use more

Stick your neck out!

Peer reviews provide value

Bigger is not better, *better* is better!

How big and deep is your hole?

Good estimates fuel good processes (and vice versa too)

Take a whiff – use your nose

Know your life cycle tolerance

Get together – more!

A Parting Reminder
(or a future seed?)

Observation: Even a turtle has to stick out its neck to make progress.

Lesson: We have plenty to learn from the rest of creation.

Helpful ants along the way . . .

I gratefully acknowledge the contributions of the following peers and friends:

Dave Leong, technical staff member at Sandia for his support in “shooting and filming” the ants. And especially for exposing himself to certain ridicule while we stared inquisitively at the ants as uninformed “scientists” pondered our purpose in life.

Laura Halbleib, a technical staff statistician at Sandia for her statistical analysis that validated intuitive inferences by applying rigorous statistical methods. Her insights and knowledge increased the reliability and usefulness of this material.

Iraj Hirmanpour, as associate with the Software Engineering Institute for his assistance in providing data from the PSP^{SM} courses. Without this data, analysis would not have been possible.

Bill Hufschmidt of the Development Support Center, a long time associate and mentor, for the information on the Function Point studies.

Thank you too for your *ant*endance and *ant*tention today!