Part II

Evaluating Process Improvement

MOTIVATING EXAMPLE

Piloting Pair Programming in Trialweb

- MedSoft managers read a research report that claimed Pair Programming (PP) increased code quality by 20% at a penalty of 15% increase in total development effort
- MedSoft decides to pilot PP in Trialweb to gauge the cost effectiveness of PP in own context
  - Does PP increase quality? By how much?
  - Does PP reduce productivity? By how much?
  - What's the net impact on bottom line?

Management Constraints

- At any given time, no more than 75% of the team will practice PP
- If PP has an unexpected and unacceptable negative impact, it will be withdrawn
- Quality implications: only care about impact on major bugs and their downstream cost
- Usage of PP will vary from release to release

MedSoft's Development Process (Releases)

- Trialweb will be developed and delivered incrementally in 6 releases employing Strategy B:
  - 3 custom releases with the lead customer
  - followed by 3 general releases
  - followed by a maintenance period
- Each release
  - has variable scope
  - is composed of XP-like prioritized stories drawn from a product backlog
  - has a dedicated pre-deployment (exploratory) functional testing phase performed in an operational environment

MedSoft's Development Process (Stories)

- Each story
  - is estimated in XP-like story points (Pts) on a difficulty scale of 1-5
  - is accompanied by a set of acceptance tests that are executed frequently throughout development
  - is considered implemented only when it passes all of its acceptance tests
  - is separately tested at the end of the release in an operational environment (functional testing)

Medsoft's Measurement Process (Productivity)

- Development effort is tracked for each project (includes release-end functional testing)
- Velocity of projects are tracked for different risk categories
  - Projects in different risk categories have comparable characteristics: complexity, team size, technical uncertainty
  - Velocity is adjusted for team size: Pts per (p)erson-time
  - Average benchmark velocities are published for each risk category
  - Trialweb's risk category: External-Medium (business as usual)
    - Team size: 6-12 developers
    - Complexity: medium
    - Benchmark velocity: 5.6 Pts/p-week

MedSoft's Measurement Process (Quality)

- Maintenance effort is tracked separate from development effort
- Defects discovered during release-end functional testing and in the field are logged in an issue backlog
  - Each found defect is traced to a particular release
  - Each defect is classified as either minor or major
  - The total effort spent to fix a defect is recorded before the associated issue is closed
  - If an issue is open, the underlying defect is still outstanding

Questions

- How should MedSoft pilot PP within Trialweb under the constraints set by management
- After the project is completed, how should MedSoft evaluate PP's effectiveness to decide whether to spread the process across the organization or not?
- Based on the data collected, what's the expected contribution of PP's organization-wide adoption to company's value?

PART II KEY QUESTIONS & ASSUMPTIONS

Key Questions

- Is it possible to assess cost-effectiveness of new process improvement initiatives?
- What strategies can organizations employ in assessing cost-effectiveness?
- What are the process measurement needs of cost-effectiveness evaluation?
- What are the caveats of process measurement?
- What are the components of cost-effectiveness?
- How can cost-effectiveness be quantified?
- How is cost-effectiveness interpreted?
- What are the components of cost-effectiveness?

Key Assumption

It's possible to measure productivity and quality

$\Rightarrow$ Reasonable proxies exist

- Reasonably *reliable* over the long term
- Reasonably *meaningful* at least in restricted contexts
- Uncertain, but uncertainty can be quantified

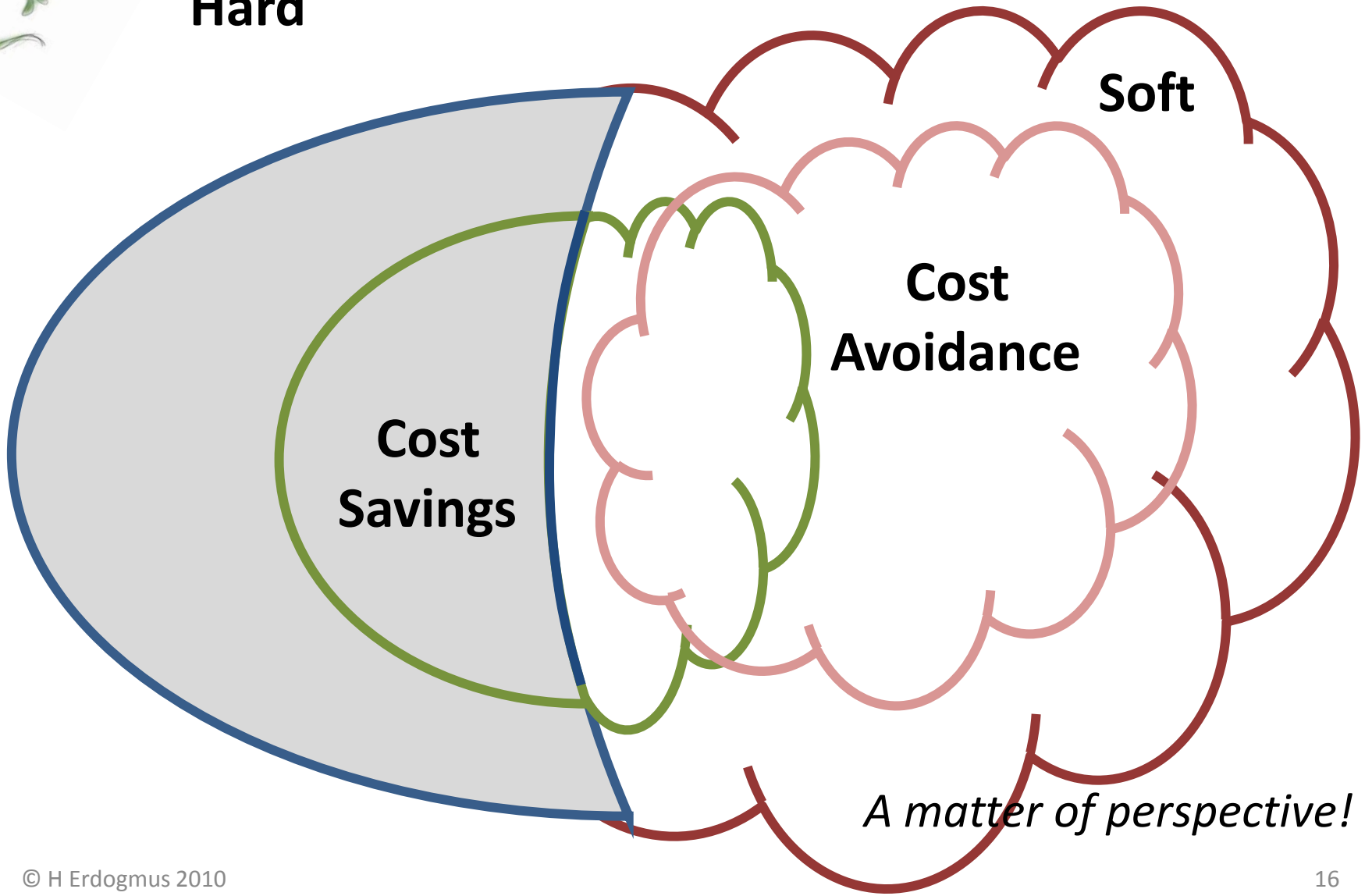
TERMINOLOGY REVIEW

- Project vs. Initiative
- Hierarchy of Benefits
- Production vs. Rework
- Quality, Issue
- Schedule, Effort
- Nominal vs. Real Productivity
- Calendar vs. Resource Productivity
- Cost Effectiveness
- Measure, Metric, Indicator

Project vs. Initiative

- Project: activity requiring specific resources to produce a desired outcome with specific benefits;
  - has a beginning, but not necessarily an end
  - *Typically targets revenue generation*
- Initiative: a way of working that requires available resources to be used in a specific way
  - an activity spanning projects (a cross-project project)
  - has a beginning, but usually continues indefinitely
  - *Typically targets cost savings or cost avoidance in the form of improved efficiency*

Benefits Hierarchy



Soft to Hard Benefits

Cost Avoidance

?

Cost Savings

Over time

Whether soft or hard, should not matter from a bottom line impact perspective so long as long-term/final impact is the same...

Production vs. Rework

Work that adds value

- resulting in a possibly partial or imperfect, but useful output
- output may have *issues* requiring resolution

Work classified as waste

- resolving any outstanding issues in work that adds value
- any related downstream corrective action
- extra effort expended to remedy imperfections

Is this an artificial separation?

- *Separation is dependent on context*
- *Must decide what constitutes waste in a given context*

Quality and Issue

- Issue:
  - an identifiable, undesirable state of an activity's output
  - incurs latent cost or prevents anticipated benefits from being realized

- Quality: extent with which an activity's output is free of issues

Schedule and Effort

- Schedule: duration of an activity in elapsed time
- Unit schedule: time unit used to measure schedule
- Effort: resource requirement of an activity, measured in person-time, where time is expressed as a multiple of unit schedule

Cost and Benefit

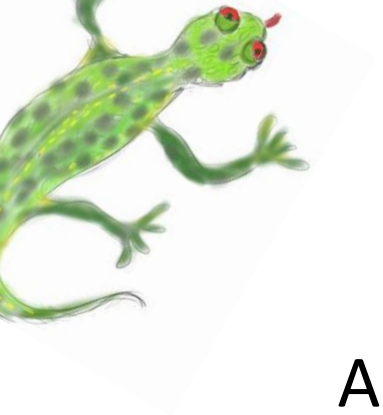
- Cost: often same as effort
  - alternatively monetary worth of effort

- Benefit: worth of output
  - hard or soft
  - can be cost savings or cost avoidance
  - expressed in monetary or equivalent-effort terms

Comparable Activities

- Two activities are (economically) comparable if
  - **their output is measurable through a common unit**
  - **the measurements are interpreted in the same manner and on the same scale**

- For the comparison to be meaningful, additional conditions may be required:
  - **Compared activities have a common objective**
  - **Compared activities are independent**
  - **Compared activities have similar resource requirements**
  - **Compared activities have similar risk**
  - **Compared activities' quality is measured through a common unit**



Productivity

An activity's average speed of production

Output/Input

or

Output/Effort

(*when* Input = Labor)

Nominal vs. Real Productivity

Nominal Productivity

- an activity's average speed of gross production
- short-term
- excludes rework

Real Productivity

- an activity's average speed of real production
- long-term
- includes rework

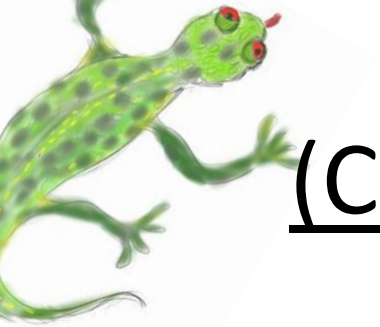
Calendar vs. Resource Productivity

Calendar Productivity

- speed of production measured in output per unit schedule
- *how fast?*

Resource Productivity

- speed of production measured in output per unit effort
- *how efficient?*



(Cost-, Economic) Effectiveness

an activity's overall performance based on its productivity and quality characteristics

Measure, Metric, Indicator

- Measure: a low-level, elementary quantity that we can estimate, count, observe, or measure about an activity

- Metric: a mid-level, derived measure;
  - a few measures' combination/aggregation
  - has an intuitive interpretation

- Indicator: a high-level, composite metric that supports decision making
  - things that you would display on a dashboard

Absolute vs. Relative Indicator

- Absolute: not normalized with respect to scale
- Relative: normalized with respect to scale

EVALUATING A PROCESS IMPROVEMENT INITIATIVE

Steps Involved in Cost-Effectiveness Evaluation

- Identify and gather baseline data
- Identify a performance model
  - Define output, quality, production, rework
- Devise an assessment strategy
- Map benefits
- Collect new data
- Calculate indicators: range estimates vs. point estimates
- Enumerate decision-making contexts
  - Identify unaccounted/intangible factors and their impact
- Analyze/interpret indicators in each context
- Pick the best fitting context, make a decision

SHOULD MEDSOFT IMPROVE ITS CODE INSPECTION PROCESS?

Key Objectives

- Identify the proper performance model for a process improvement initiative
- Devise a strategy to assess cost effectiveness
- Calculate Net Value, ROI, Nominal Productivity, and Real Productivity in a cost savings context
- Illustrate proper use of absolute vs. relative indicators
- Illustrate utility of different indicators in different decision-making situations

Case

- MedSoft has a pre-integration code inspection process that is performed on all newly developed piece of software classified as “critical” (required to have a near-zero defect rate)
- A new piece of research shows the defect-finding performance of code inspections can substantially improve if the inspection cycle is repeated by a new inspection team
- MedSoft wants to gauge whether it’s cost-effective to introduce reinspections for “critical” software

Case inspired by Biffl, Freimut, Leitenberger, 2001

MedSoft's Existing Inspection Process

- MedSoft's inspection sessions on average require 50 person-hours (p-hr) each
- Defects in "critical" software are classified at 4 severity levels
- If found post-integration, on average:
  - A Severity 0 defect takes an extra 0.1 p-hr to fix
  - A Severity 1 defect takes an extra 1 p-hr to fix
  - A Severity 2 defect takes an extra 8 p-hr to fix
  - A Severity 3 defect takes an extra 80 p-hr to fix
- All defects found in "critical" software must be fixed before release
- A single inspection session on average finds 50% of all defects in each of severity level

Baseline Data

from 10 randomly selected inspection sessions

Total defects and estimated savings per found defect according to severity

Defect Severity	0	1	2	3	
# Defects	17	31	29	9	
Saved Effort	0.1	1	8	80	p-hr/defect

Inspection

Defect Severity	0	1	2	3
# Defects Found	8	14	14	4
# Defects Missed	9	17	15	5

Discovered post-integration

Invested Effort ---------- 146 p-hr

Performance Model

Two comparable activities involving potential downstream rework:

- Inspection only
- Inspection + Reinspection

where

- Output measure: # defects found (this is what we want)
- Quality measure: # defects missed (Issue = missed defect)
- Production: effort spent by inspection team on inspection prep + inspection meeting
- Rework: extra downstream effort necessary for fixing missed defects

Assessment Strategy

- MedSoft decides to evaluate the performance of reinspections using “critical” code that has previously been inspected, integrated, tested, and released
- It uses data from 10 randomly selected inspection sessions performed on this piece of code
- A team of inspectors who have not participated in the original inspections reinspect the code to apply the methodology suggested by new research to find any additional defects missed by the original inspection teams; the results are recorded

New Data

Reinspection and effort results for 10 sessions

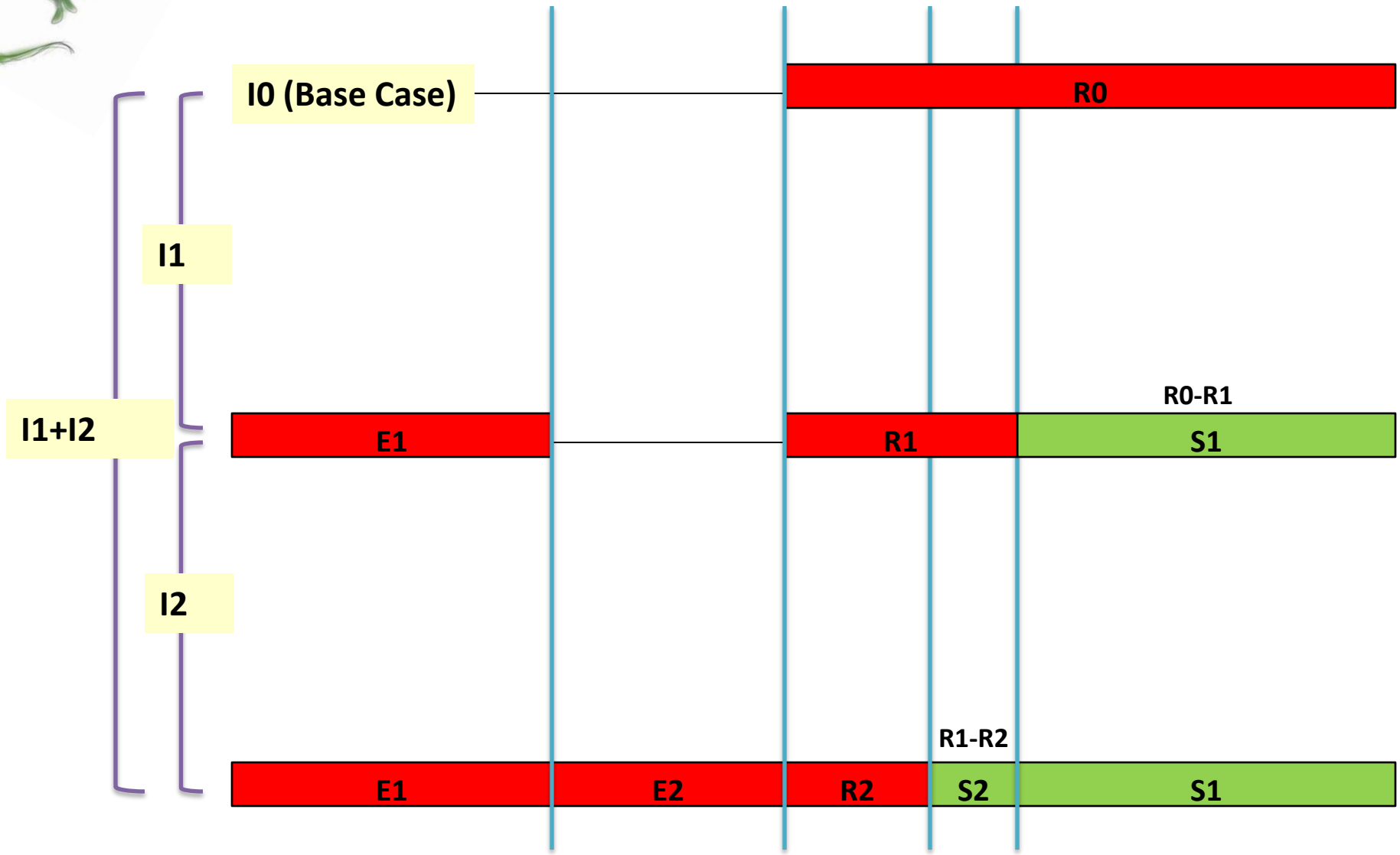
Reinspection				
Defect Severity	0	1	2	3
# Defects Found	2	4	4	2
# Defects Missed	7	13	11	3
Invested Effort	135 p-hr			

Discovered during reinspection

Benefits Mapping

I1: Inspection
I2: Reinspection
I0: No Inspection

E: Tot. I1 or I2 Effort
R: Rework
S: Savings



Level I Analysis: Relative Effectiveness of Reinspections

Inspection Only (I_1)

Defect Severity	0	1	2	3	All	
# Defects Found	8	14	14	4	40	
# Defects Missed	9	17	15	5	46	
Invested Effort					146	p-hr
Saved Effort	0.8	14	112	320	447	p-hr
Net Value					301	p-hr
ROI					**2.1**	
Productivity (nominal)					0.27	defects/p-hr

14 *severity-2 defects* x
8 *p-hr per severity-2 defect*

Total for all found defects

Benefit – Cost

Net Value / Cost

All found defects / Cost

Reinspection (I_2)

Defect Severity	0	1	2	3	All	
# Defects Found	2	4	4	2	12	
# Defects Missed	7	13	11	3	34	
Invested Effort					135	p-hr
Saved Effort	0.2	4	32	160	196	p-hr
Net Value					61	p-hr
ROI					0.5	
Productivity (nominal)					0.09	defects/p-hr

Looks like by all measures,
reinspections are not as efficient as inspections,
but still provide positive gains

Level 2 Analysis: Inspection vs. Inspection+Reinspection

Inspection Only (I1)						
Defect Severity	0	1	2	3	All	
# Defects Found	8	14	14	4	40	
# Defects Missed	9	17	15	5	46	
Invested Effort					146	p-hr
Saved Effort	0.8	14	112	320	447	p-hr
Net Value					301	p-hr
ROI					2.1	
Productivity (nominal)					0.27	defects/p-hr

Reinspection (I2)						
Defect Severity	0	1	2	3	All	
# Defects Found	2	4	4	2	12	
# Defects Missed	7	13	11	3	34	
Invested Effort					135	p-hr
Saved Effort	0.2	4	32	160	196	p-hr
Gain					61	p-hr
ROI					0.5	
Productivity (nominal)					0.09	defects/p-hr

Inspection + Reinspection (I1+I2)		
Invested Effort	281	p-hr
Net Value	362	p-hr
ROI	1.3	
Productivity (nominal)	0.19	defects/p-hr

Inspection alone has better ROI and Nom. Prod., but Inspection+Reinspection has better Net Value!
Now what?

- *For one-off evaluation with mutually exclusive alternatives in a closed world:* absolute indicators (Net Value) override relative indicators (ROI, Prod.)
- *When reporting systematic effects:* only relative indicators (ROI, Prod.) are meaningful

Level 3 Analysis: Real Productivity

Inspection Only (I1)						
Defect Severity	0	1	2	3	All	
# Defects Found	8	14	14	4	40	
# Defects Missed	9	17	15	5	46	
Invested Effort					146	p-hr
Saved Effort	0.8	14	112	320	447	p-hr
Net Value					301	p-hr
ROI					2.1	
Productivity (nominal)					0.27	defects/p-hr
Rework effort	0.9	17	120	400	538	p-hr
Productivity (real)					0.13	defects/p-hr
Inspection + Reinspection (I1+I2)						
Invested Effort					281	p-hr
Saved Effort					362	p-hr
ROI					1.3	
Productivity (nominal)					0.19	defects/p-hr
Rework effort	0.7	13	88	240	342	p-hr
Productivity (real)					0.14	defects/p-hr

(40+46)/(146+538)

Real indicators are preferred to nominal ones *unless* rework (long-term) costs are highly uncertain!

Not much difference between real productivities!

Operating under limited resources?
Possibly additional hidden cost of poor quality?

Are ROI and Net Value Nominal or Real?

Inspection Only						
Defect Severity	0	1	2	3	All	
# Defects Found	8	14	14	4	40	
# Defects Missed	9	17	15	5	46	
Invested Effort					146	p-hr
Saved Effort	0.8	14	112	320	447	p-hr
Net Value					301	p-hr
ROI					2.1	
Productivity (nominal)					0.27	defects/p-hr
Rework effort	0.9	17	120	400	538	p-hr
Productivity (real)					0.13	defects/p-hr

- Net Value and ROI both depend on Tot. Savings
- Here Tot. Savings and Rework Effort are inversely related (their sum is constant)
- Thus ROI and Net Value both capture to some extent cost of poor quality in this case

Net Value and ROI are real indicators in this context!

Different Indicators are Differently Meaningful in Different Decision-Making Situations

1) Dedicated or unlimited resources: highest profitability with maximum effectiveness
   - Inspection+Reinspection, because estimated net value is higher

2a) Interchangeable or limited resources: short-term effects, best allocation of resources for a future project
   - Inspection Only because it represents better use of resources now (not sure whether all defects will be found and will need to be repaired) – *instead of reinspection, perform inspection on a different part of the system, perhaps on non-critical code!*

2b) Interchangeable or limited resources: long-term effects, all defects eventually found and fixed, maximize organization-wide defect detection performance
   - Indifferent because real productivities are close *-- if reinspection has hidden opportunity cost, that hidden cost will work against reinspection; if defects have hidden cost, that cost will work in favor of reinspection*

      - *safety critical system – situation 1*
      - *desktop application with 20 releases - situation 2*
         - *It's in release 15 (short life span ahead): situation 2a*
         - *It's in release 1 (long life span ahead): situation 2b*

MEASURING PRODUCTIVITY & QUALITY

Process Measurement Challenge

- Can't measure output or quality in software development in any direct and universally meaningful way

  - Rely on for *internally meaningful* **proxies**

Meaningful Measurement

A good goal for improvement!

- Internally meaningful – comparable inside:
  - single project
  - project category
  - portfolio
  - organization
  - domain

Inherently difficult!

- Externally meaningful – universal validity, comparable across a sector, all types of software

Measuring Productivity

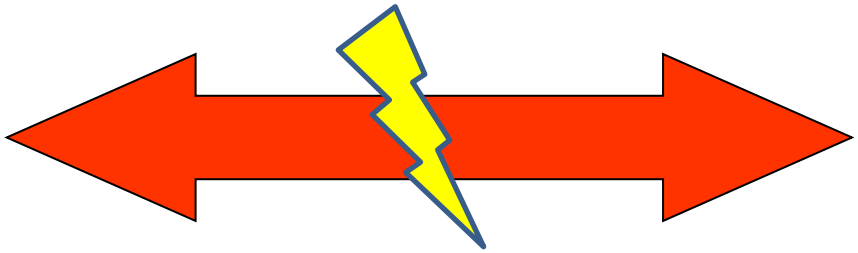
Productivity = Output per Unit Input

Must measure this

What constitutes a meaningful output proxy?

- Size?
- Functionality?

Productivity Measurement Tradeoff



Granularity & Uniformity

Coarse/Low

Fine/High

Correlation with Functionality

High

Low

Function points
Use cases
User stories
Scenarios
Acceptance tests passing
Story points
Covered/Adjusted SLOCs*
SLOCs delivered

**excludes untested code or adjusted for duplication*

Low Uniformity and Coarse Granularity

Low uniformity => counting difficulties; may require a weighting scheme to normalize unequally sized units (large stories get a small weight, small stories get a large weight, etc. – *also apply to quality measures*)

Coarse granularity => instability due to low values, high variance (averaging out effects may be absent)

Program A	Program B
1 user story	2 user stories
1000 SLOCs	2000 SLOCs

- For which measure can you state with more confidence that one of the programs has produced approximately twice as much output as the other?

Measuring Quality

Desirable attributes of software artifacts themselves

Attributes that affect usage or downstream effort: *issues* reported during operation *or* interfering with actual use; *deployed* defects, bugs, *or* faults; reliability, performance; also: actual *ease of maintenance, extension, and evolution*

Internal Quality vs. External Quality

This is important only to the extent that it correlates with external quality, i.e., as a proxy for external quality!

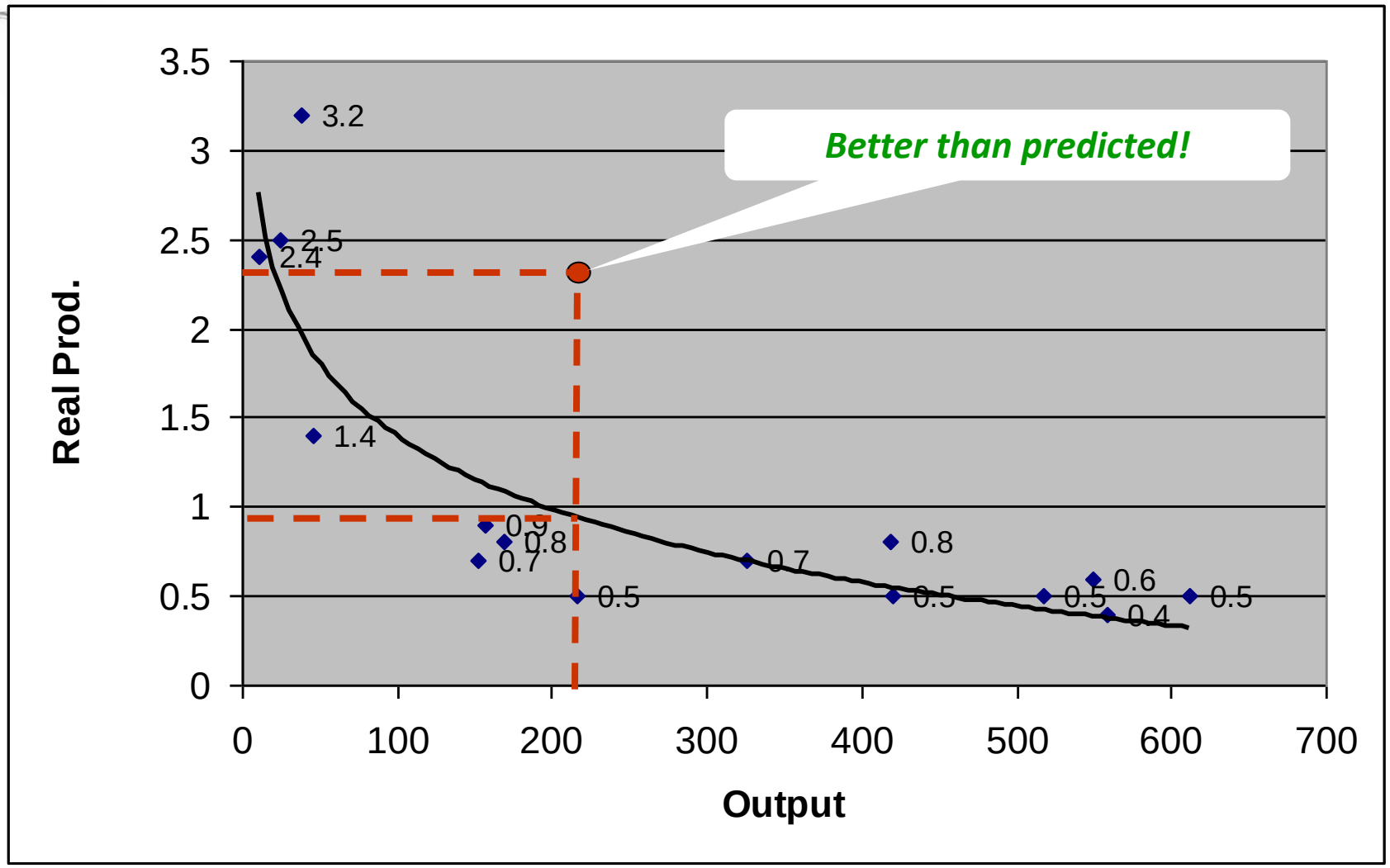
This is what ultimately counts!

Measurement Pitfalls: Relative Indicators, Nonlinearity and Scale

Kitchenham, Jeffery, Connaughton article, "Misleading metrics and unsound analysis", IEEE Software, Mar/April 2007

- Compare only similar projects with each other
- Do not rely on benchmarks based on average productivity over a portfolio of projects with mixed characteristics (size, complexity, risk, domain)
- Effort depends on project scale (size, output)
- For cost prediction purposes, use nonlinear regression models: Effort = *f*(Output) where *f* is non-linear
- Evaluate productivity through visualizing the relationship between its components (Effort and Output)
- **Gauge meeting productivity targets by comparing predicted productivity to actual productivity**

Predicted vs. Actual Productivity

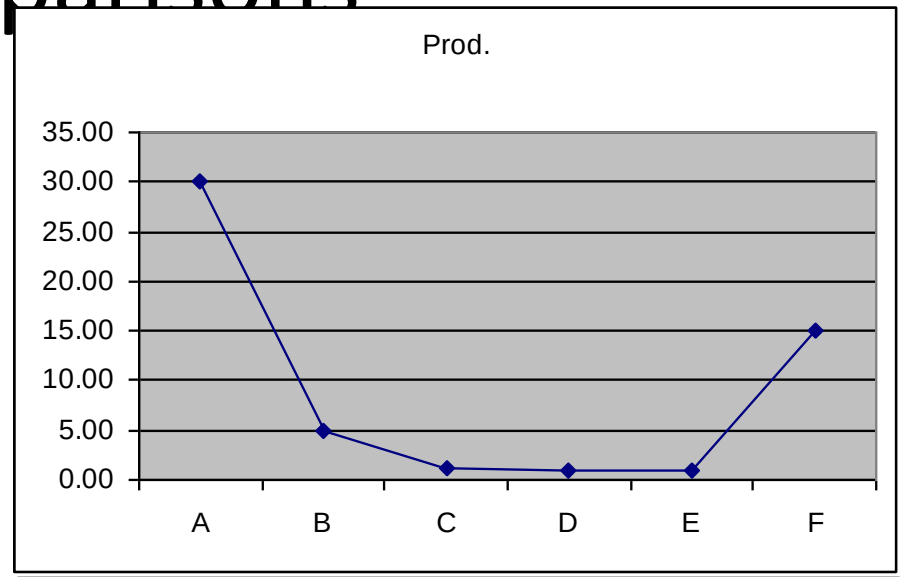


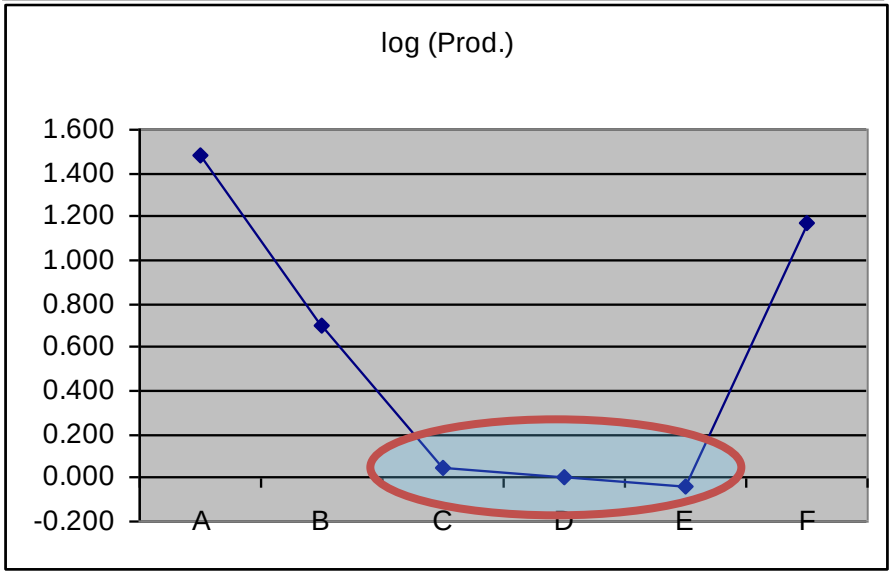
Be careful about regressing a metric that derives from a dependent variable against that dependent variable!

Visualization Problems in Comparisons

Kitchenham et al: *Do not use ratios (relative indicators) to gauge productivity*

- Ratios may hide scale effects, obscure functional forms when you plot them; differences among low values may not be visible!
- *If this is just a visualization problem*
  - *plot X/Y along a log scale; i.e. plot log X – log Y rather than X/Y to see hidden small-scale differences.*
  - *Group values in bins representing different scales; plot bins separately*





REAL PRODUCTIVITY: WHY IS IT A COST-EFFECTIVENESS INDICATOR?

Real Productivity as a Breakeven Value

- Suppose:
  - we are able to measure the output of an activity by a reasonable proxy, but we don't know how that output translates into monetary value
  - Assume: Benefit of Activity = its *Production Value*

Ω v ← Hypothetical, Unknown, Average

Production Value = Output x Unit Value

(Currency) = (Output) x (Currency/Output)

(Currency) ⇐ (Output)

Net Value of Activity

- **Total Cost** = Total Effort (E) x Unit Salary (S)

- **NPV** = **-Total Cost + (Production Value)**

  (Currency) = (Person-time) x (Currency/Person-Time)

Breakeven Unit Value (BUV)

$$\mathbf{BUV} = \min \{ \mathbf{v} \mid \mathbf{NPV} \geq 0 \} = f(\mathbf{E}, \boldsymbol{\Omega}, \mathbf{S})$$

smallest UV that makes NPV break even

Solve **v** for **NPV** = 0 $\Rightarrow$ $\mathbf{BUV} = \mathbf{S} \cdot \mathbf{E} \cdot / \boldsymbol{\Omega}$

Real Productivity = $\mathbf{RP} = \boldsymbol{\Omega}/\mathbf{E}$

BUV = S/RP (Currency/Output)

Small BUV => increased profit margin, breaking even easy

Large BUV => decreased profit margin, breaking even difficult

Smaller is better, more profitable!

Breakeven Multiple (BM)

- S can be assumed to be invariant in the same context
- What fraction of S is the BUV? How many BUVs can you squeeze into the unit salary?

BM = S/BUV = RP (Output/Person-Time)

RP is more robust than NPV for assessing systematic benefits!

SHOULD MEDSOFT ADOPT PAIR PROGRAMMING?

Key Objectives

- Demonstrate how to pilot a new approach in a live context under specific management constraints
- Identify measurement needs for piloting new approaches
- Gauge cost effectiveness based on pilot results
- Illustrate usage of baseline data for sanity checking and meaningful comparison

Baseline Data: Literature

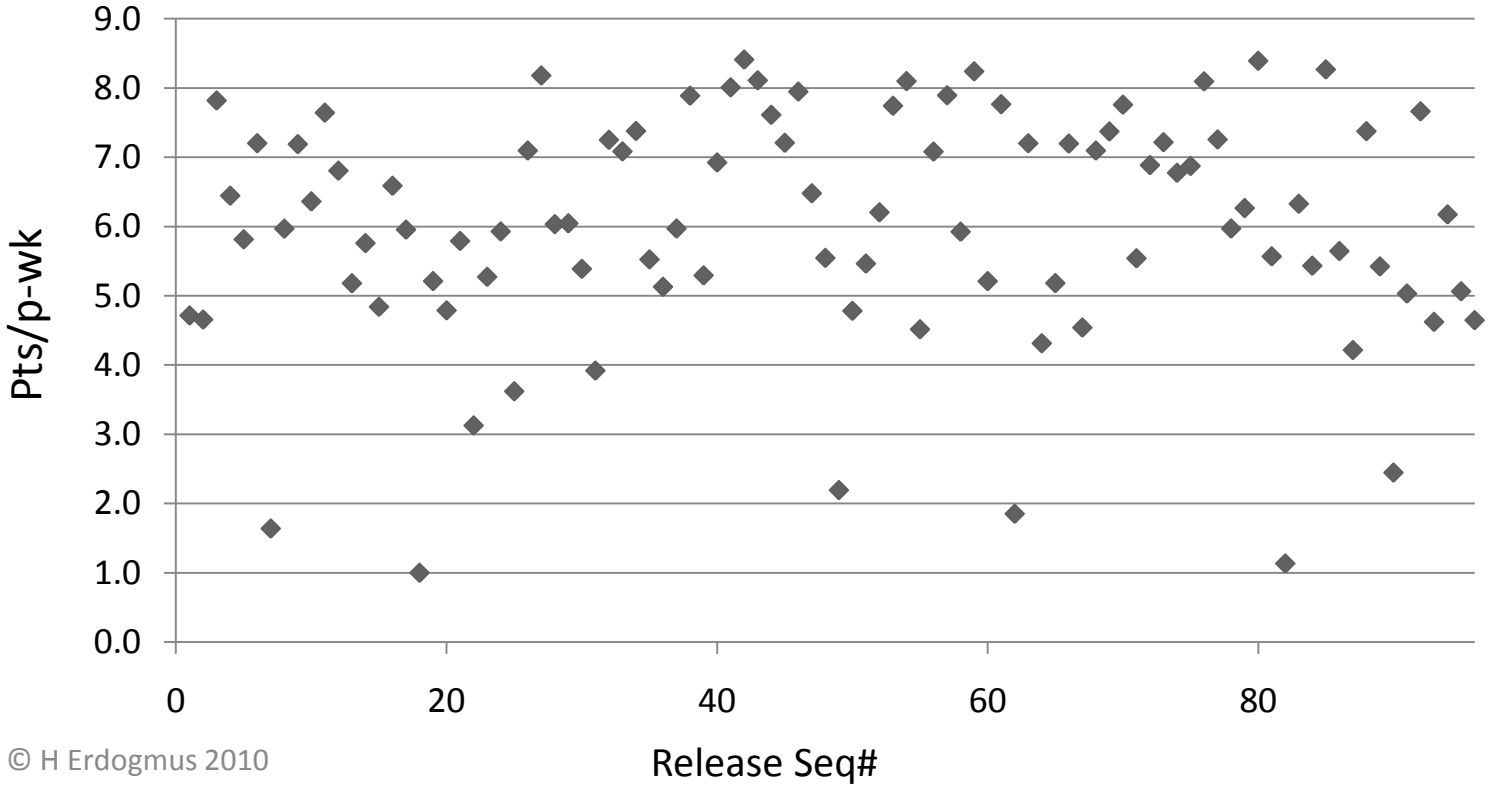
- Claimed benchmark to be tested: 20% quality increase in return for a 15% productivity penalty

?

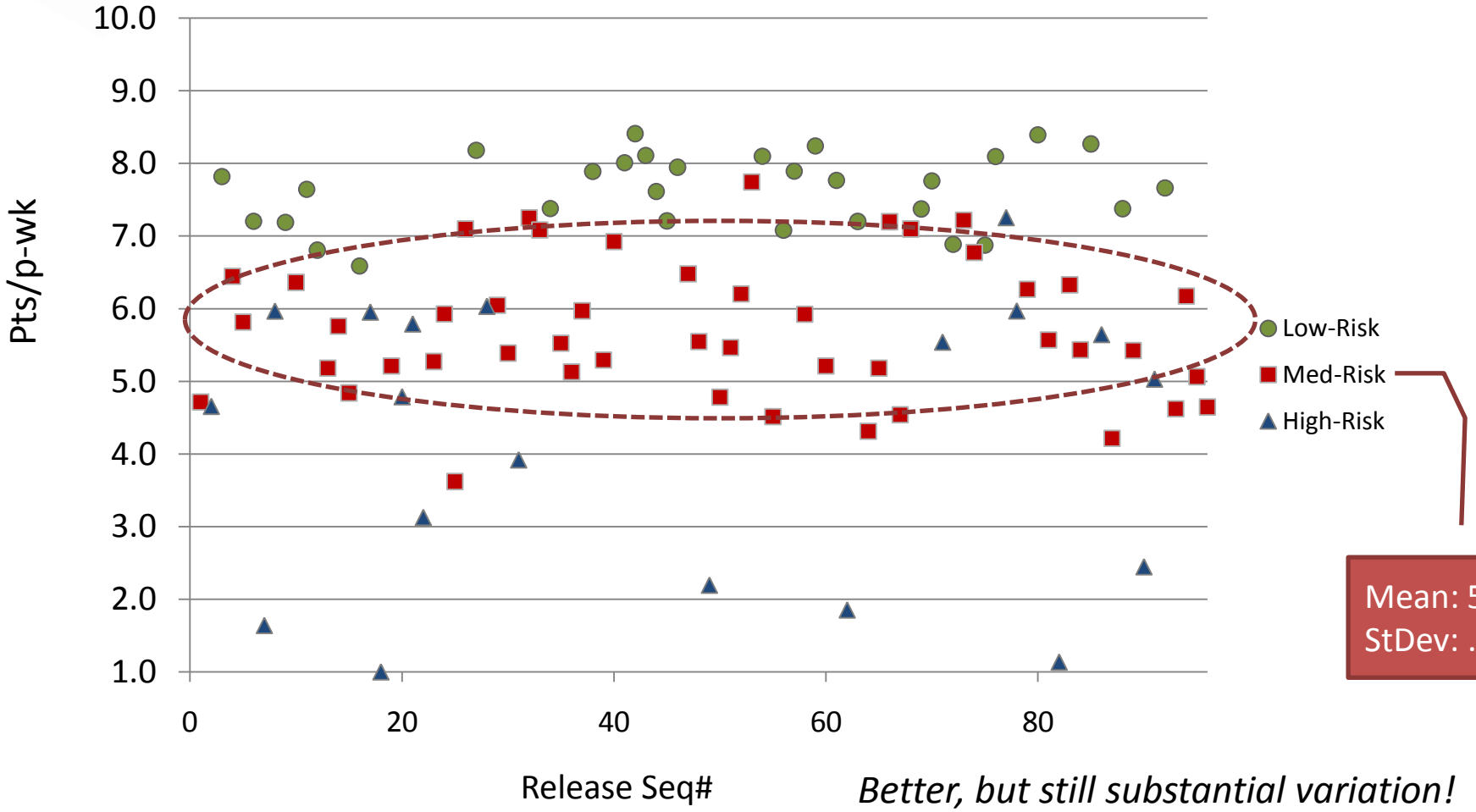
Baseline Data: Velocity

- Normalized and quality-corrected velocity (Pts/p-wk) from 96 releases over 12 projects (8-12 person teams)

Too much variation!



Baseline Data: Velocity for Comparable Projects



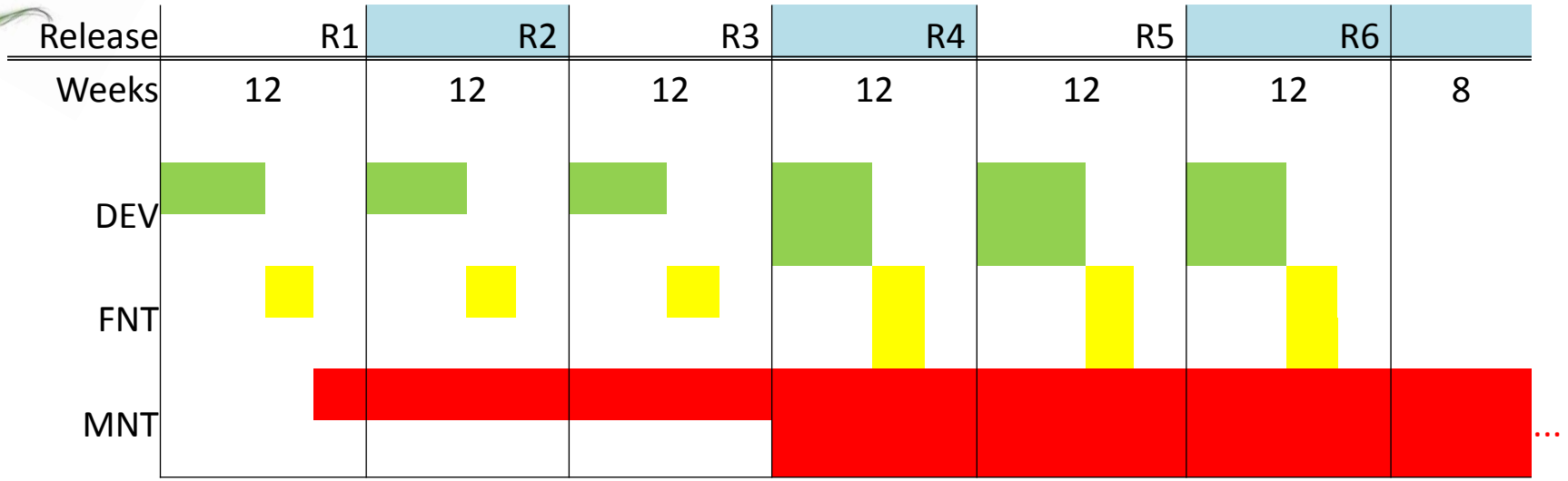
Better, but still substantial variation!

Trialweb PP Pilot

Recall management constraints

- At any given time, no more than 75% of the team will practice PP
- If PP has an unexpected and unacceptable negative impact, it will be withdrawn
- Only impact of major bugs and their downstream cost is a concern
- Usage of PP will vary from release to release

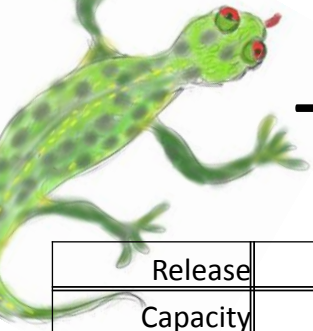
Trialweb PP Pilot
Resource Load Plan



Team Size	8	8	8	12	12	12	6
Pair	0	3	0	3	0	4	2
Solo	8	2	8	6	12	4	2
Unit	8	5	8	9	12	8	4
%PP	0%	75%	0%	50%	0%	67%	67%

Release	R1	R2	R3	R4	R5	R6	

PP training in R1



Trialweb PP Pilot: Release Data

	Release	R1	R2	R3	R4	R5	R6		p-hr		
	Capacity	3840	3840	3840	5760	5760	5760	1920			
	DEV + FNT Effort	2646	2906	2427	4536	4098	4612	0			
	Minor MNT Effort	384	480	426	523	640	523	384			
									Done	**Extra**	***Total***
Major Bugs	R1	18	15	6	3	1	0	0	**43**	**0**	***43***
	R2	-	8	15	4	1	0	0	**28**	**0**	***28***
	R3	-	-	12	18	9	2	1	**42**	**0**	***42***
	R4	-	-	-	6	7	4	0	**16**	**1**	***17***
	R5	-	-	-	-	22	12	6	**37**	**3**	***40***
	R6	-	-	-	-	-	9	8	**14**	**3**	***17***
	Total	**18**	**23**	**33**	**31**	**40**	**27**	**15**	**180**	**7**	***187***
Major MNT Effort	R1	810	630	234	114	42	0	0	**1830**	**0**	***1830***
	R2	-	304	675	156	39	0	0	**1174**	**0**	***1174***
	R3	-	-	504	684	342	78	39	**1647**	**0**	***1647***
	R4	-	-	-	270	315	152	0	**737**	**46**	***783***
	R5	-	-	-	-	924	540	252	**1716**	**139**	***1855***
	R6	-	-	-	-	-	378	312	**690**	**148**	***838***
	Total	**810**	**934**	**1413**	**1224**	**1662**	**1148**	**603**	**7794**	**333**	***8127***
	Velocity (Pts)	304	684	547	797	836	961	**0**	**4129**	**0**	***4129***

Trialweb PP Pilot: Efficiency

Release	R1	R2	R3	R4	R5	R6	
PP%	0%	75%	0%	50%	0%	67%	
Velocity (Pts)	304	684	547	797	836	961	
Production Effort	2646	2906	2427	4536	4098	4612	(p-hr)
Total Effort	4476	4080	4074	5319	5953	5450	(p-hr)
Efficiency	59%	71%	60%	85%	69%	85%	

Efficiency = Production Effort / Total Effort

Trialweb PP Pilot: Nominal Productivity

Release	R1	R2	R3	R4	R5	R6	
PP%	0%	75%	0%	50%	0%	67%	
Velocity (Pts)	304	684	547	797	836	961	
Production Effort	2646	2906	2427	4536	4098	4612	(p-hr)
Total Effort	4476	4080	4074	5319	5953	5450	(p-hr)
Efficiency	59%	71%	60%	85%	69%	85%	
Nom. Productivity	4.6	9.4	9.0	7.0	8.2	8.3	(Pts/p-wk)

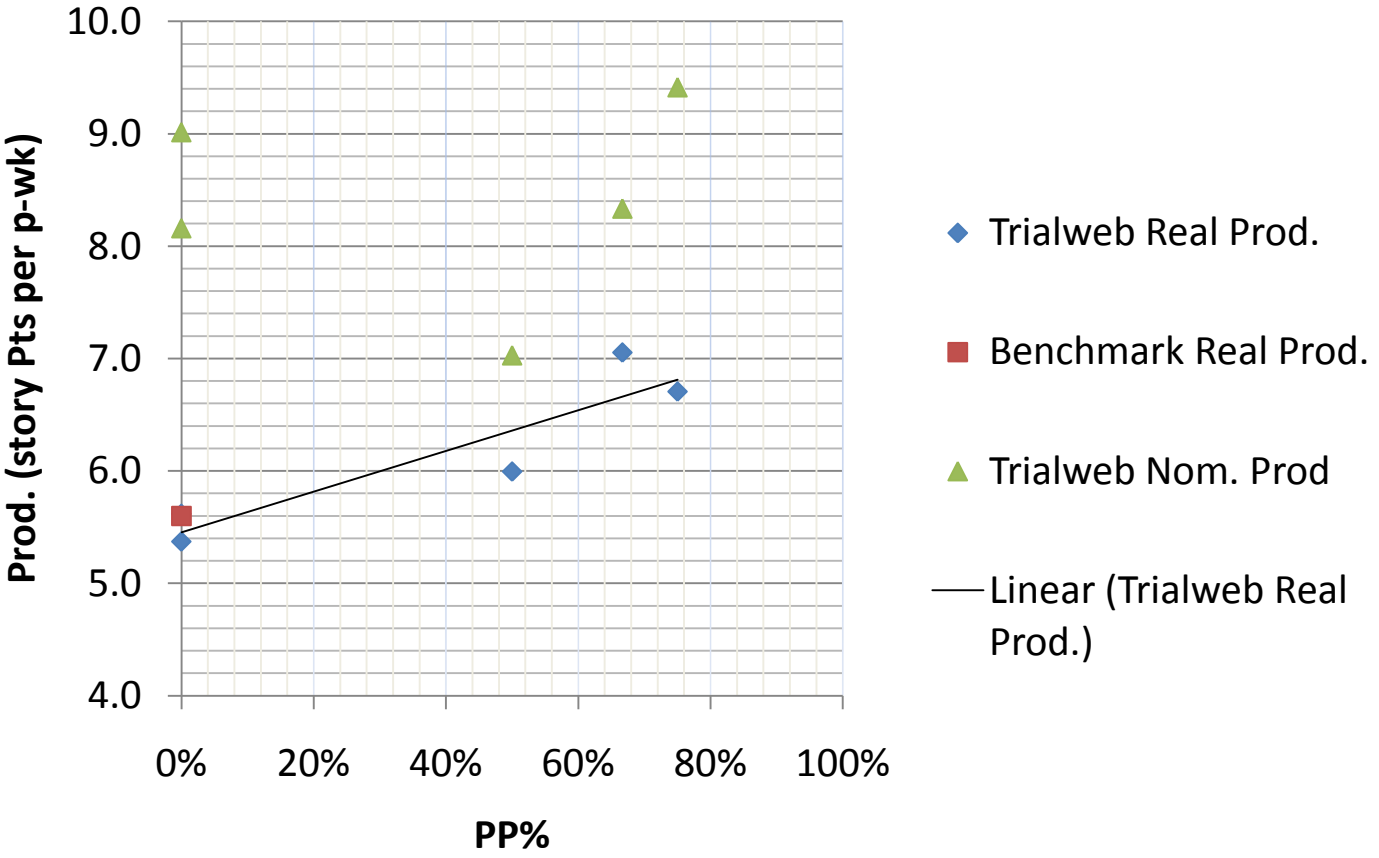
Nom. Productivity = Output / Production Effort

Trialweb PP Pilot: Real Productivity

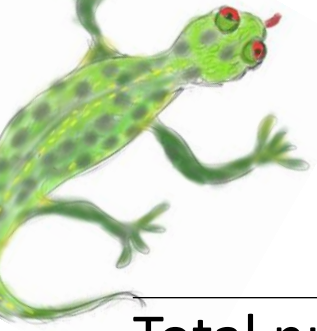
Release	R1	R2	R3	R4	R5	R6	
PP%	0%	75%	0%	50%	0%	67%	
Velocity (Pts)	304	684	547	797	836	961	
Production Effort	2646	2906	2427	4536	4098	4612	(p-hr)
Total Effort	4476	4080	4074	5319	5953	5450	(p-hr)
Efficiency	59%	71%	60%	85%	69%	85%	
Nom. Productivity	4.6	9.4	9.0	7.0	8.2	8.3	(Pts/p-wk)
Real Productivity	2.7	**6.7**	**5.4**	**6.0**	**5.6**	**7.1**	(Pts/p-wk)

Real Productivity = Output / Total Effort

Trialweb PP Pilot: Visualization



Min. Improvement from Benchmark: 7%
Max. Improvement from Benchmark: 26%

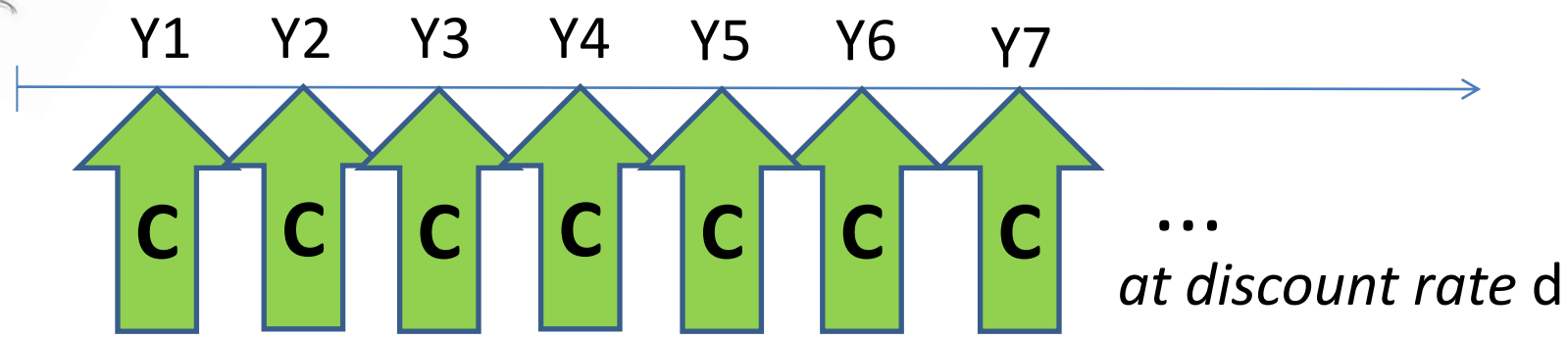


Scaling Up Systematic Effects: NPV and ROI of PP Adoption

Total number of developers in MedSoft	300		
Average salary of MedSoft developer	$125K/year		
PP Training Cost for TrialWeb Team	$30.00K	NPV in PI risk category at 20%	ROI in PV Terms
Total PP Training Cost	$750.00K		
Savings @ Min Improvement	$2,635K/year =	$15,062 K	20
Savings @ Max Improvement	$9,733K/year =	$57,645 K	77

300*125*7%

Calculating Perpetual NPV

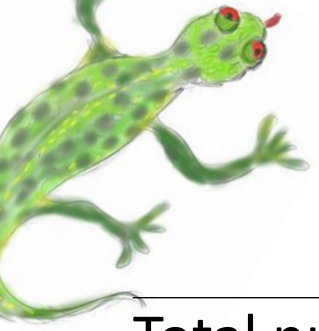


$$PV = C/(1 + d) + C/(1 + d)^2 + C/(1 + d)^3 + \ldots$$
$$= C[1/(1+d) + 1/(1 + d)^2 + 1/(1 + d)^3 + \ldots] = C \cdot \Sigma$$

But $\Sigma = [1/(1+d)][1 + \Sigma]$

Then $\Sigma = (1 + d)/d$

$$PV = C \cdot (1 + d)/d$$



2635*(1+.2)/.2 - 750

15062/750

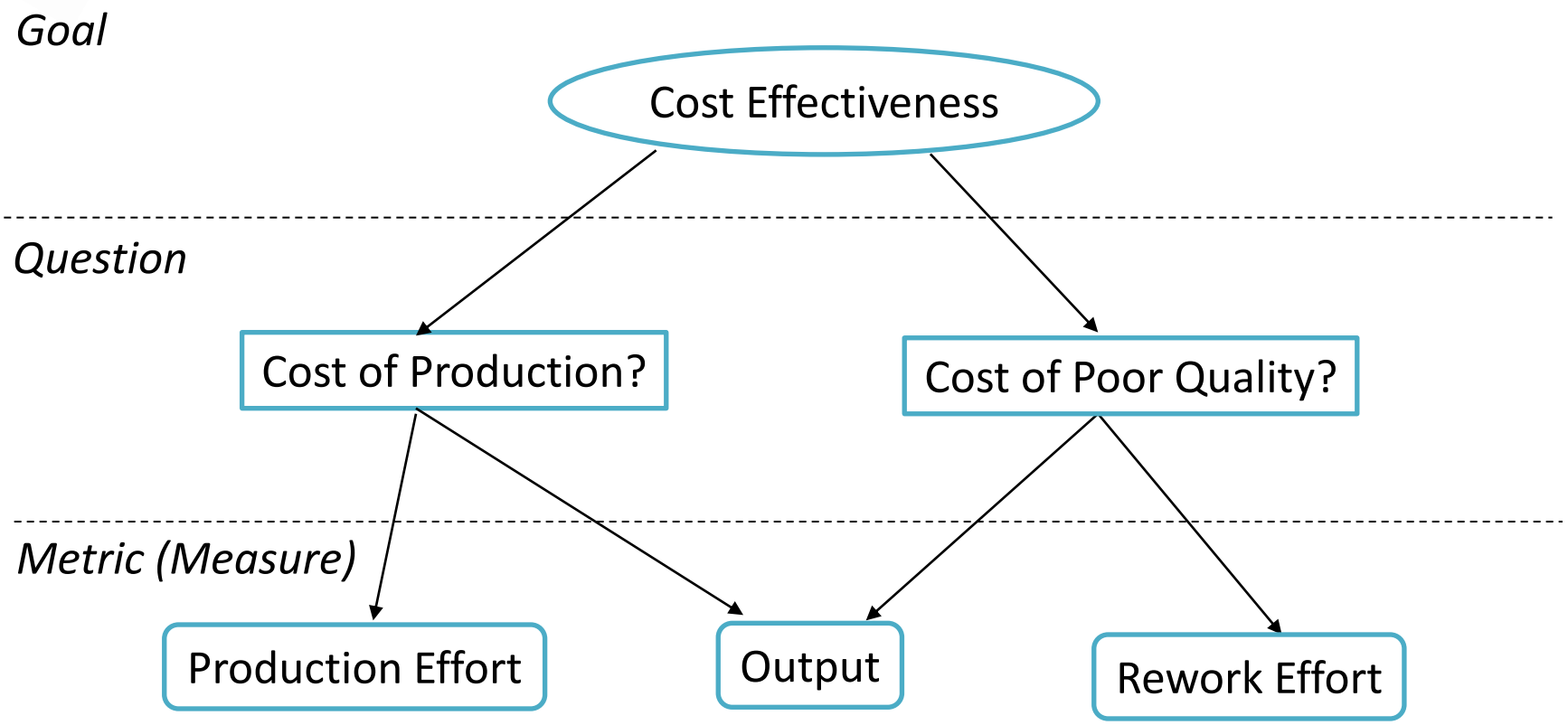
Total number of developers in MedSoft	300		
Average salary of MedSoft developer	$125K/year		
PP Training Cost for TrialWeb Team	$30.00K	NPV in PI risk category at 20%	ROI in PV Terms
Total PP Training Cost	$750.00K		
Savings @ Min Improvement	$2,635K/year =	$15,062 K	20
Savings @ Max Improvement	$9,733K/year =	$57,645 K	77

What Next?

Based on these results it would make sense for MedSoft to continue rolling out PP gradually inside the organization and keep monitoring its cost-effectiveness

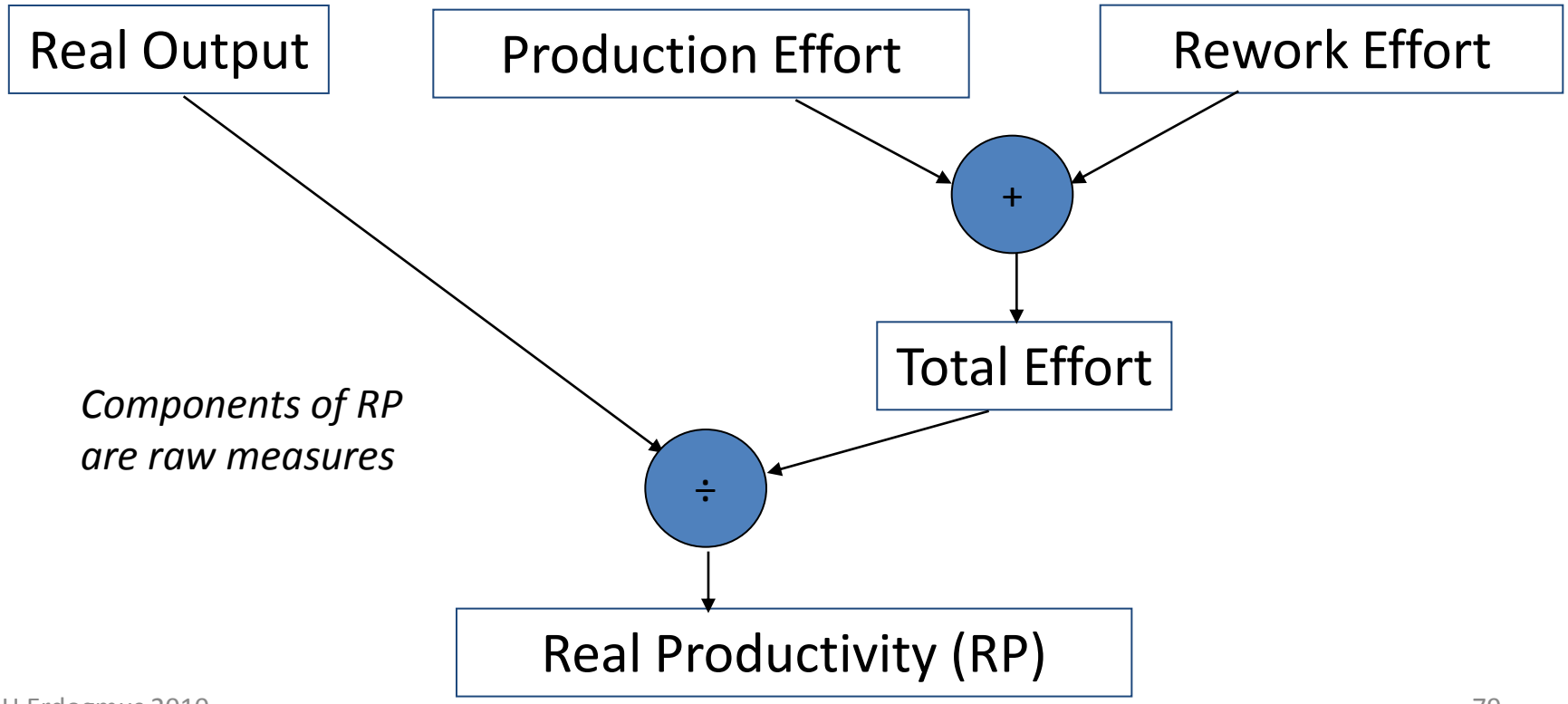
BREAKING DOWN REAL PRODUCTIVITY

General Form of Cost-Effectiveness Models



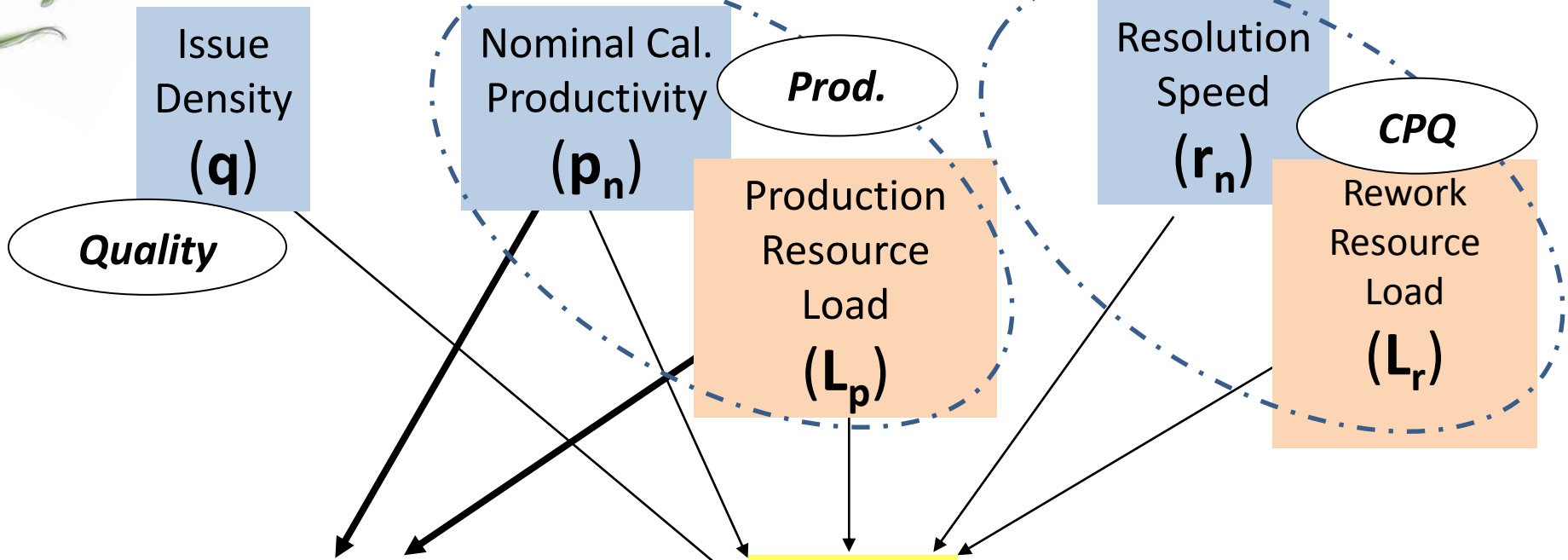
Direct Calculation of RP

… from absolute measurements of production effort, rework effort, and output



Components of RP are raw measures

Compositional RP Model



Blue components relative metrics!

$$RP = p_n \cdot e / L_p$$

$$e = \frac{L_p \cdot r_n}{(L_p \cdot r + L_r \cdot q \cdot p_n)}$$

Resource Load under Variable Salary Scales (Variable Scale Case)

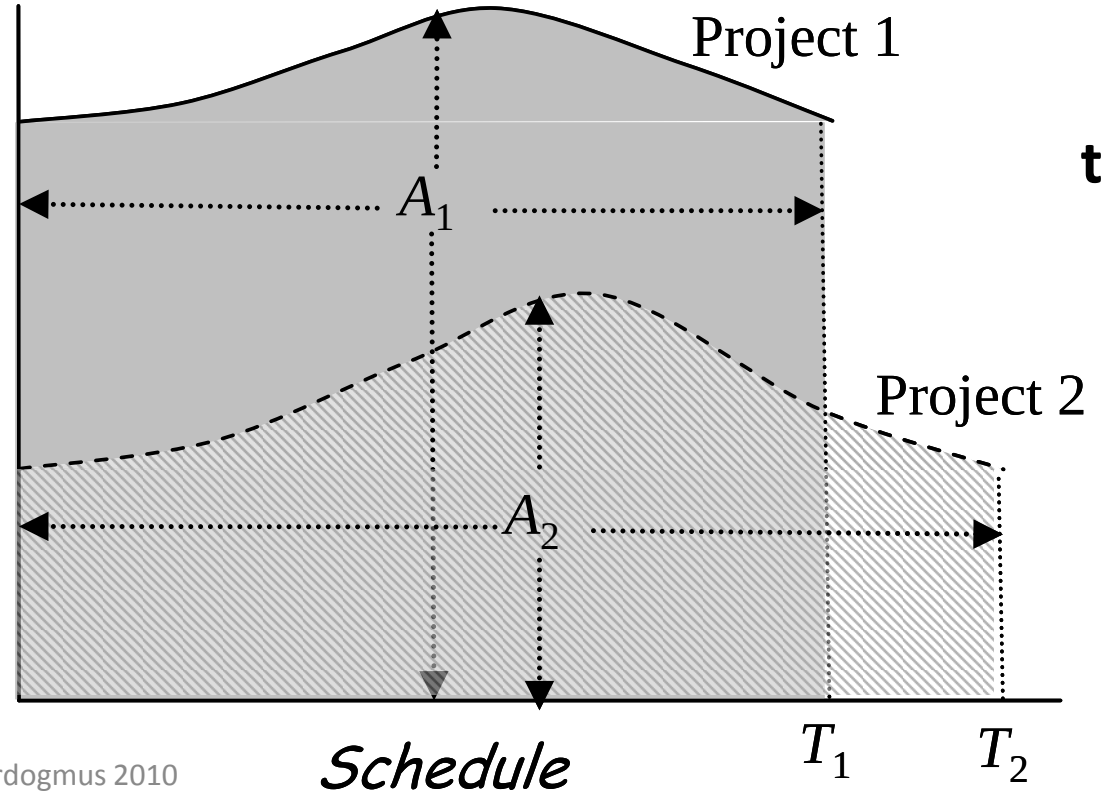
Divide total load by a base salary

	Project 1 salaries (3 members)	Project 2 salaries (4 members)
Base salary = $100K	$80K	$80K
	$100K	$120K
	$120K	$120K
		$150K
Total salary	$300K	$470K
Resource Load	$L_1 = 3.0$ persons	$L_2 = 4.7$ persons

Resource Load Varying with Schedule (Variable Load Case)

Salary load (persons)

$$L_1 = A_1/T_1;\ L_2 = A_2/T_2$$



Compute the average load

Cost of Poor Quality (CPQ)

- Sensitivity analysis of RP is difficult with 5 parameters
- Express efficiency and real productivity in terms of two aggregate components:
  - 1st component: represents CPQ
  - 2nd component: nominal productivity in resource (effort) terms rather than in calendar (schedule) terms

Calculating CPQ

Average rework effort per unit output

$CPQ = L_r \cdot q / r_n$ (person-time/output)

q: issue density (issues/output)

L_r: rework resource load (persons)

r_n: resolution speed in calendar terms (issues/hour)

(r_n/L_r is resolution speed in resource terms)

Nominal Resource Productivity

Nominal productivity in resource (effort) terms

$$p_{n,res} = p_n / L_p \quad \textbf{(output/person-time)}$$

p_n: Nominal (Calendar) Productivity (output/time)

L_p: production resource load (persons)

Efficiency and RP in terms of CPQ

Efficiency

$$e = 1/(1 + CPQ \cdot p_{n,res})$$

Real Productivity

$$RP = e \cdot p_{n,res}$$

Main Components of RP

1. Cost of Poor Quality

2. Nominal Productivity

IMPROVING MEDSOFT'S REQUIREMENTS PROCESS?

Key Objectives

- Demonstrate the calculation of RP from its components
- Demonstrate sensitivity of RP to Cost of Poor Quality

Case

- Clinicians using MedSoft's online services complain about clunky UIs and poor user experience
- A substantial part of MedSoft's maintenance costs are attributed to resolving UI-related issues to keep customers happy
- MedSoft is considering to modify its development workflow to include an activity based on a User-Centered Design (UCD) approach to resolve customer complaints and save maintenance costs

Assessment Strategy

- MedSoft forms 3 teams to prototype 2 sets of Trialweb functionality, F1 and F2, of similar complexity
  - Team T0 has 4 analysts trained in UCD
  - Team T1 has 2 UI developers
  - Team T2 has 3 UI developers
- 4 case studies are conducted
  - Case A: T0 is coupled with T1 to implement F1
  - Case B: T0 is coupled with T2 to implement F2
  - Case C: T2 implements F1 alone using existing process
  - Case D: T1 implements F2 alone using existing process

Performance Model

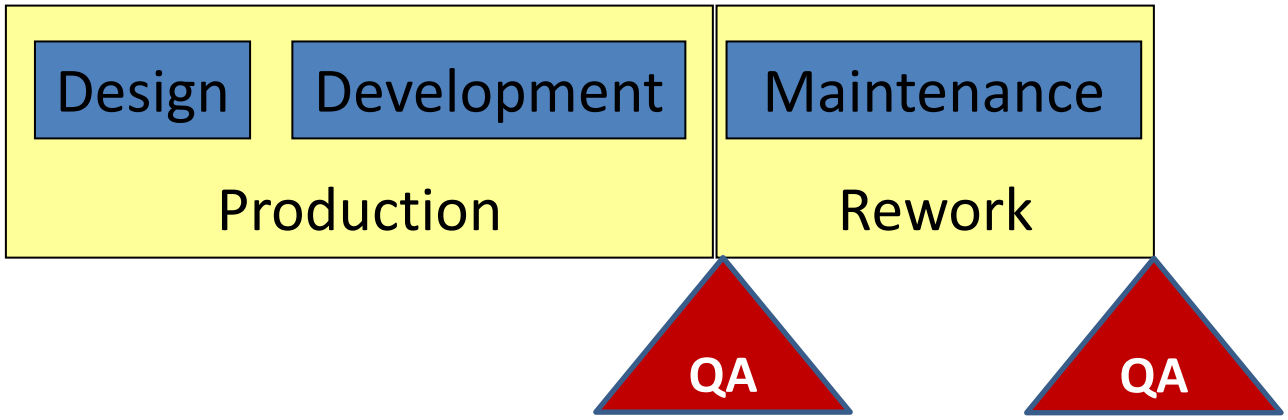
- A Quality Assessment (QA) with users is conducted after implementing the required functionality, followed by a maintenance phase to resolve any issues discovered, followed by a final QA
- Output is measured in terms of Function Points (FPs)
- Quality is measured in terms of UI defects found after initial implementation
- Cost of Poor Quality is estimated by tracking UI defects repaired during maintenance
- Schedules and resource loads for each kind of activity are tracked

Mapping

4 case studies, 2 involving UCD, where

- Output measure: Function Points (FP)
- Quality measure: UI Defects
- Production: Optional Design (UCD) + Development Effort
- Rework: Maintenance Effort to fix UI Defects

(Issue = UI Defect)



Case Study Data

	Case A	Case B	Case C	Case D	
Phase I - Design					
Design Time	7	5	0	0	hours
Design Team Size	4	4	0	0	persons
Phase II - Development					
Development Time	16	21	26	18	hours
Development Team Size	2	3	3	3	persons
Functionality implemented	15	12	7	5	FP
Major Defects Found	4	6	3	8	major defects
Minor Defects Found	6	5	11	6	minor defects
Phase III - Maintenance					
Maintenance Time	10	6	14	23	hours
Maintenance Team Size	2	2	2	2	persons
Major Defects Repaired	2	5	1	4	
Minor Defects Repaired	5	4	7	5	

UCD

No UCD

QA

Calculating RP

$RP = e \cdot p_n / L_p$

		Case A	Case B	Case C	Case D	
Output		15	12	7	5	FP
Issue Weight of Major Defect		*2*	*2*	*2*	*2*	*issues/defect*
Issue Weight of Minor Defect		*1*	*1*	*1*	*1*	*issues/defect*
Design Effort		28			0	person-hours
Development Effort		32			54	person-hours
Rework Effort		20	12	28	46	person-hours
Production Effort		60			54	person-hours
Total Effort		80			100	person-hours
Production Schedule		23			18	hours
Total Schedule		33			41	hours
Resource Load - Production	L_p	2.61	3.19	3.00	3.00	persons
Resource Load - Rework	L_r	2.00	2.00	2.00	2.00	persons
Nominal Calendar Productivity	p_n	0.65	0.46	0.27	0.28	FP/hour
Issues Found		14	17	17	22	issues
Issue Density	q	0.93				issues/FP
Issues Resolved		9		9	13	issues
Issues Outstanding		5	3	8	9	issues
Resolution Speed	r_n	0.90				issues/hour
Outstanding Rework Effort		11.1				person-hours
Efficiency	e	66%	85%	60%	41%	
Real Productivity	RP	0.16	0.12	0.05	0.04	FP/person-hour

(Des. Effort) + (Dev. Effort.)

(Des. Sched) + (Dev. Sched.)

(Prod. Effort)/(Prod. Sched.)

(Issues Resolved)/(Rework Sched.)

$$\frac{\text{(Issues Outstd.)(Rework Res. Load)}}{\text{(Resolution Speed)}}$$

(Prod. Effort)/(Tot. Effort)

Surprise!

Also substantial improvement in nominal productivity in addition to better quality

Recall: CPQ $\Rightarrow$ e $\Rightarrow$ RP

$$CPQ = L_r \cdot q / r_n \quad \text{(person-time/output)}$$

$$p_{n,res} = p_n / L_p \quad \text{(output/person-time)}$$

$$e = 1/(1 + CPQ \cdot p_{n,res})$$

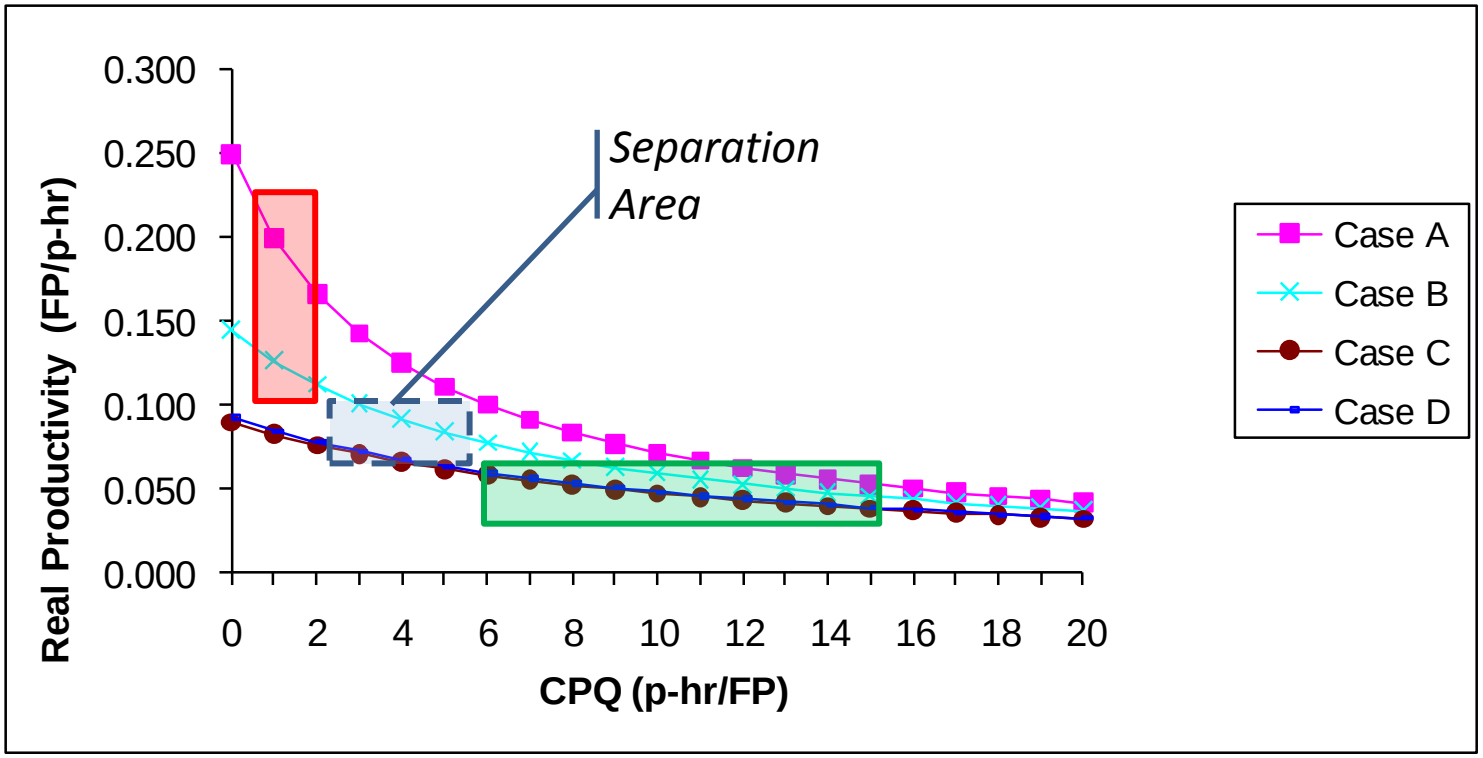
$$RP = e \cdot p_{n,res} \quad \text{(output/person-time)}$$

Impact of CPQ on RP

Cost of Poor Quality (fixed Nom. Productivity, variable Efficiency)

	Case A		Case B		Case C		Case D	
CPQ	Eff.	Prod.	Eff.	Prod.	Eff.	Prod.	Eff.	Prod.
0	100%	0.250	100%	0.145	100%	0.090	100%	0.093
1	80%	0.200	87%	0.126	92%	0.082	92%	0.085
2	67%	0.167	78%	0.112	85%	0.076	84%	0.078
3	57%	0.143	70%	0.101	79%	0.071	78%	0.072
4	50%	0.125	63%	0.092	74%	0.066	73%	0.068
5	44%	0.111	58%	0.084	69%	0.062	68%	0.063
6	40%	0.100	54%	0.077	65%	0.058	64%	0.060
7	36%	0.091	50%	0.072	61%	0.055	61%	0.056
8	33%	0.083	46%	0.067	58%	0.052	57%	0.053
9	31%	0.077	43%	0.063	55%	0.050	55%	0.051
10	29%	0.071	41%	0.059	53%	0.047	52%	0.048
11	27%	0.067	39%	0.056	50%	0.045	50%	0.046
12	25%	0.063	37%	0.053	48%	0.043	47%	0.044
13	24%	0.059	35%	0.050	46%	0.041	45%	0.042
14	22%	0.056	33%	0.048	44%	0.040	44%	0.040
15	21%	0.053	32%	0.046	43%	0.038	42%	0.039
16	20%	0.050	30%	0.044	41%	0.037	40%	0.037
17	19%	0.048	29%	0.042	40%	0.036	39%	0.036
18	18%	0.045	28%	0.040	38%	0.034	38%	0.035
19	17%	0.043	27%	0.039	37%	0.033	36%	0.034
20	17%	0.042	26%	0.037	36%	0.032	35%	0.032

Sensitivity of RP to CPQ



UCD

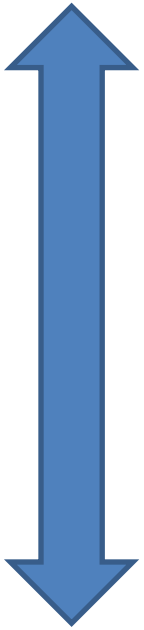
No-UCD

ROLE OF EVIDENCE IN PROCESS IMPROVEMENT

3 Levels of Evidence

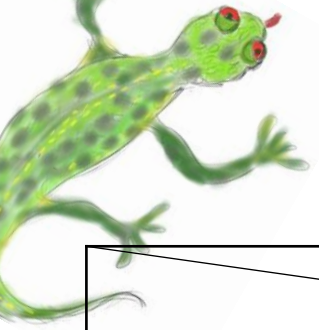
- Feasibility Confirmation
  - Strengths, Weaknesses, Opportunities, Threats (SWOT)
  - Survey
  - Sandboxing
- Anecdotal Evidence
  - Collection of observations from isolated cases
- Systematic Evidence
  - Based on aggregating credible and rich data

Weak



Strong

- Gathered from *inside* or *outside* organization
- The more risk-averse, the stronger the evidence hurdle



When Is Evidence Necessary?

Risk-averse →

Situation \ *Type of Evidence*	**Feasibility Confirmation**	**Anecdotal Evidence**	**Systematic Evidence**
Viral, self-evident isolated idea	SOME	NO	NO
Before exploring an idea	YES	NO	NO
Before exposing an idea	YES	SOME	NO
Isolated practice, piloting	YES	YES	NO
Isolated practice, gradual large-scale adoption	YES	YES	SOME
Isolated practice, small-scale adoption	YES	YES	SOME
New process, gradual large-scale adoption	YES	YES	SOME
Isolated practice, rapid large-scale adoption	YES	YES	YES
New process, small-scale adoption	YES	YES	YES
New process, rapid large-scale adoption	Not a good idea		

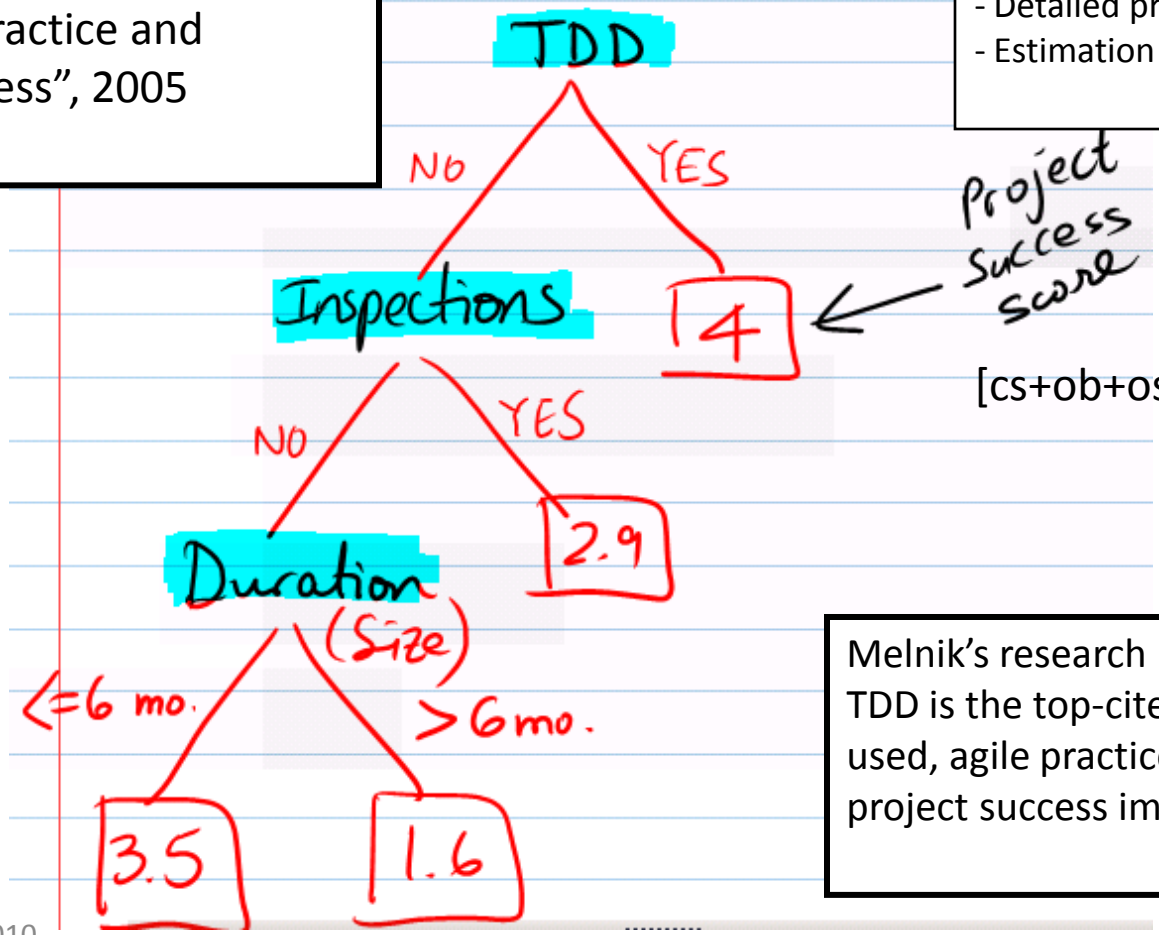
New process: whole set of practices

IS TEST-DRIVEN DEVELOPMENT READY FOR PRIME TIME?

TDD: A Frequently Quoted Practice

Cutter Consortium Survey of196 projects
Khaled El-Emam,
"Software practice and project success", 2005

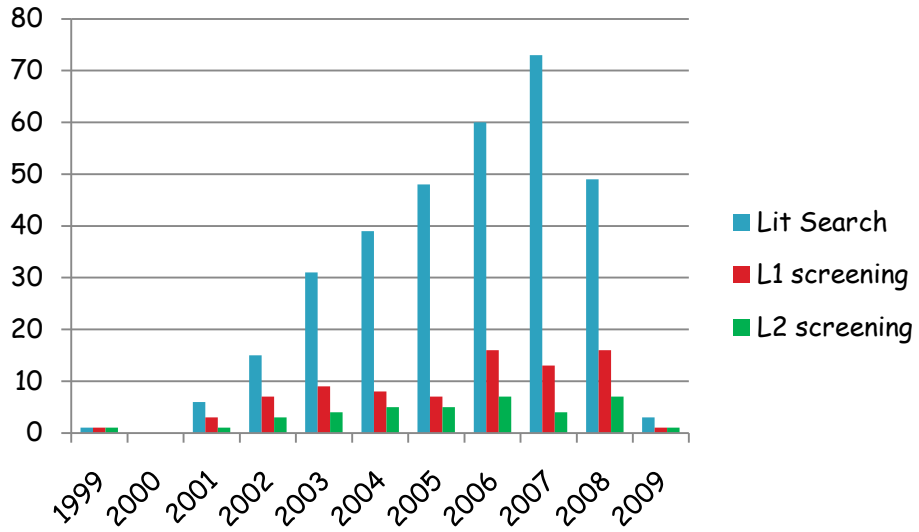
Practices that were surveyed but did not make the ranking:
- Architectural design
- Documentation
- Detailed project planning
- Estimation practices



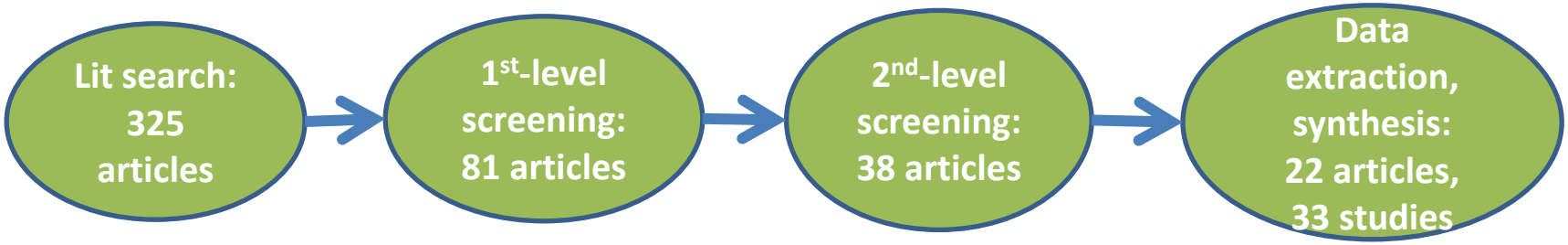
[cs+ob+os+q+p]

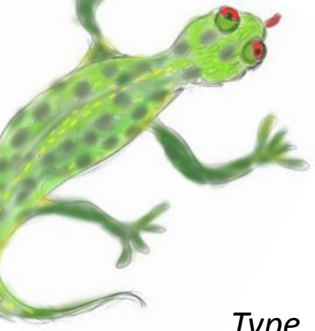
Melnik's research also indicates TDD is the top-cited, but least used, agile practice in terms of project success impact

Systematic Evidence: How Gathered?



- Systematic lit review using "case survey" method
- Exclusive focus on studies that evaluated TDD with respect to productivity & quality relative to a more traditional approach





Quality (All Studies)

Type	BETTER	NO-DIFF	WORSE	
Controlled Experiment	✓	✓ ✓ ✓	✓ ✓	**6**
Pilot Study	✓ ✓ ✓ ✓ ✓ ✓	✓ ✓		**8**
Industrial Use	✓ ✓ ✓ ✓ ✓ ✓	✓		**7**
ALL STUDIES:	**13**	**6**	**2**	**21**

Hypothesis: TDD improves external quality

Intuition: Developers work on well-specified tasks, use frequent regression testing, and can quickly find errors due to changes.

- Overall, pilots & industrial data support the hypothesis (experiments inconclusive)

and % tests passed,
defects, defect density, defects per test,
QA effort,
change density, % preventive change

Quality (Best Studies)

Type	BETTER	NO-DIFF	WORSE	
Controlled Experiment	✓	✓ ✓ ✓	✓ ✓	6 ***5***
Pilot Study	✓ ✓ ✓ ✓ ✓ ✓	✓ ✓		8 ***7***
Industrial Use	✓ ✓ ✓ ✓ ✓ ✓	✓		7 ***1***
ALL STUDIES:	13	6	2	21
HIGH-GRADE STUDIES:	***5***	***6***	***2***	***13***

Hypothesis: TDD improves external quality

Intuition: Developers work on well-specified tasks, use frequent regression testing, and can quickly find errors due to changes.

- Overall, pilots & industrial data support the hypothesis (experiments inconclusive)
- BUT evidence is contradictory when less rigorous studies excluded.

Moderate evidence in favour of TDD

Quality (More Info)

- In 3 studies:
  - external quality is more consistent with TDD
  - external quality likely depends on testing effort rather than exact process dynamics (test-first vs. test-last)
- In 2 studies:
  - more test effort with TDD did not translate to better average external quality, but increased minimum quality

Productivity (Nominal?, All Studies)

Type	BETTER	NO-DIFF	WORSE	
Controlled Experiment	✓ ✓ ✓	✓		**4**
Pilot Study	✓ ✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓	✓ ✓ ✓ ✓	**14**
Industrial Use	✓	✓	✓ ✓ ✓ ✓ ✓	**7**
ALL STUDIES:	**10**	**6**	**9**	**25**

Hypothesis: TDD incurs a productivity penalty

Intuition: Steep learning curve, overhead of writing and managing tests,

- Experiments favor TDD, pilots mixed, and industrial studies favor control

code productivity, feature productivity,
maintenance productivity,
development effort, maintenance effort

Productivity (Nominal?, Best Studies)

Type	BETTER	NO-DIFF	WORSE	
Controlled Experiment	✓ ✓ ✓	✓		4 2
Pilot Study	✓ ✓ ✓ ✓ ✓ ✓	✓ ✓ ✓ ✓	✓ ✓ ✓ ✓	14 11
Industrial Use	✓	✓	✓ ✓ ✓ ✓ ✓	7 1
ALL STUDIES:	10	6	9	25
HIGH-GRADE STUDIES:	**6**	**4**	**4**	**14**

Hypothesis: TDD incurs a productivity penalty

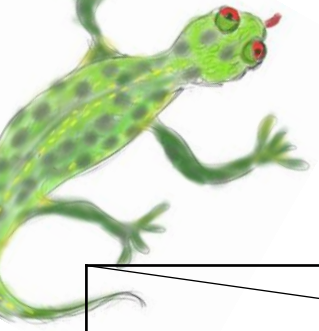
Intuition: Steep learning curve, overhead of writing and managing tests

- Experiments favor TDD, pilots mixed, and industrial studies favor control
- Same results regardless of study rigor
- Discrepancy likely due to representation/flavor of TDD

TDD doesn't have a consistent negative effect

Summary of TDD Evidence

- Moderate systematic quality benefits
- No evidence that it systematically suffers a productivity hit
  - Short term? Productivity probably suffers
  - Long term? Quality benefits may compensate for short-term productivity hit => No **real** productivity hit
- Which dimension (quality or productivity) is overriding?
  - Quality: moderately compelling evidence
  - Productivity: not sufficient info



What Should MedSoft Do? (Critical Software)

Type of Evidence / *Situation*	**Feasibility Confirmation**	**Anecdotal Evidence**	**Systematic Evidence**
Viral, self-evident isolated idea	SOME	NO	NO
Before exploring an idea	YES	NO	NO
Before exposing an idea	YES	SOME	NO
Isolated practice, piloting	YES	YES	NO
Isolated practice, gradual large-scale adoption	YES	YES	SOME
Isolated practice, small-scale adoption	YES	YES	SOME
New process, gradual large-scale adoption	YES	YES	SOME
Isolated practice, rapid large-scale adoption	YES	YES	YES
New process, small-scale adoption	YES	YES	YES
New process, rapid large-scale adoption	Not a good idea		

PART II KEY MESSAGES

- Identify and separate what constitutes value-adding activity (Production) from what constitutes waste (Rework)
- Track Production and Rework separately

- Improvement approaches that leverage productivity operate on (increase) Nominal Productivity

- Improvement approaches that leverage on quality operate on (reduce) Cost of Poor Quality

Use **relative** indicators when

- Reporting, tracking, and comparing systematic effects of process improvement
- Making decisions under flexible allocation of limited resources

Use **absolute** indicators when

- Reporting organization-wide impact of process improvement or impact of process improvement on specific projects
- Making decisions under dedicated, discretionary allocation of unlimited resources

Prefer **real** indicators over **nominal** indicators to gauge cost-effectiveness

- Real indicators capture long-term or lifecycle effects, nominal indicators don't
- Real indicators capture Cost of Poor Quality; nominal indicators don't

Make sure to include measures that help determine/estimate CPQ in your measurement program

When introducing new approaches

- Review evidence reported in the literature and evaluate its strength
- Conduct low-cost experiments (pilots, case studies) to gauge cost-effectiveness in own context
- Choose a speed of adoption commensurate with the complexity of the approach, organizational culture (including risk-averseness of the organization)

When measuring productivity and quality

- Be aware of the tradeoffs involved in using different measures
- Select output and quality measures most meaningful for the context
- If necessary, use multiple measures to cross-validate and triangulate

When interpreting results focus on behavioural properties

- Conduct sensitivity analysis
- Conduct breakeven analysis
- Provide range estimates (vs. point estimates)

Summary of Cost-Effectiveness Indicators

	(A)bs. or (R)el.	Costs	Hard Benefits	Soft Benefits	Time Value	Sys. Effects
NV	A	Yes	Yes	Yes	No	No
ROI	R	Yes	Yes	Yes	No*	Yes
NPV	A	Yes	Yes	Yes	Yes	No
Real Prod.	R	Yes	No	Yes	No	Yes

*Unless PV of benefits used in denominator