Using Simulation to Investigate Requirements Prioritization Strategies

Dan Port, Alexy Oklov (UH)
Tim Menzies (WVU)
ASE'2008


Manifesto for Agile Software Development
http://agilemanifesto.org/
Manifesto for Agile Software Development
We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
AG: agile-based
PB: plan-based
That is, while there is value in the items on the right, we value the items on the left more.
Kent Beck
Mike Beedle
Arie van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas
Downloads
Clear

This paper

- Specifically:
  - With very little machinery, we can characterize:
    - when PB (plan-based) is better/worse than AG (agile)
    - If PB or AG or XYZ is appropriate for your particular project
- More generally:
  - Our simulator is so very, very, very simple.
  - So, why…
    - Years of grand-standing about polar extremes?
    - Don't we see more automated process debates?

Related work

- COCOMO [Boehm'81&00]: hard-wired into 2 dozen variables
  - What about the concepts not mentioned in COCOMO?
- Search-based SE approach [Harmon'04].
  - optimization techniques from operations research and meta-heuristic search (simulated annealing and genetic algorithms)
  - Seeks near-optimal solutions to:
    - complex over-constrained SE models
    - Or simpler COCOMO-based models (Menzies et.al [ASE'07])
  - SBSE too complex for this requirements study
- Elaborate process simulations (e.g. [Raffo])
  - detailed insight into an organization
  - Hard to tune (e.g. Raffo's Ph.D. model, 2 years tuning effort)
  - Raffo: one large model for all questions
    - Our approach: one very small model per question

Model(λ , σ)

- λ =requirements discovery: rate of new requirements
  - Requirements += Poisson(λ)
- σ =requirements volatility: rate of requirements changing value
  - Value += max(0, value + N(0,σ))
- Steps though 2 ≤ I ≤ 6 iterations of requirements review
  - B= base requirements at iteration one (max=25)
  - Early stopping probability of 1/(maxI^0.33) = 55%
  - Requirements unimplemented at each phase: 20%
- Requirements
  - Value R_x : min_value(30) … max_value(500)
  - Cost R_x: min_cost(1) … max_cost(100)
    - Assumed to be nonvolatile

End development time is unknown

Cao, Ramesh, IEEE software 2008

Experiments with volatile costs not insightful

No inter-requirement dependencies

Two kinds of "iterations"

- *Project iterations*
  - Every so often, pause to consider what to do next
  - At each at pause, deliver Version1, version2, version3,….
- *Value iterations*
  - Every so often, the value of our requirements change.
  - Assume that after $N
    - There is a pause, and the value of each requirement is reassigned.
  - $N = (*total cost of base reqs*) / num_iters
- For three of our simulations (AG, AG2, hybrid)
  - One value iteration for each project iteration
- For conventional plan-based prioritization
  - Only one project iteration
  - But numerous value iterations

Agile Prioritization (AG)

- Requirements are prioritized at the beginning of each iteration
  - Requirements are retired, highest value first
- Many, but not all, requirements discovered at first iteration
  - Selected randomly B = 30% ≤ 40*N(0,1) ≤ 70%
- Initialization
  - R= Determine num_req
  - B = number "base requirements" (those known in iteration 1)
  - AG_heap = {1,2,….,B, B+1 ,…R}
  - AG_plan = {1,2,….,B}
- Simulation. For each project iteration:
  - Sort AG_plan on value, implement top 80%
  - AG_plan= remaining 20% of AG_plan + Poiss(λ) items of AG_heap

Cao, Ramesh, IEEE software 2008

- *A local search, so prone to local maxima*
- *Ignores cost*

AG2

- Same as AG
  - But sort on value / cost

Hybrid Prioritization (HY)

- Same as AG2 but….
- Sort AG_plan by value/cost,
  - Prune those with value/cost < α
    - α = (∑ remaining values) / (∑ remaining costs)

- *HY is a local search, prone to local maxima*
- *But α limits our exploration of dead-ends*

Plan-based Prioritization (PB)

- Requirements are prioritized before iteration 1
  – Using highest (value/cost)

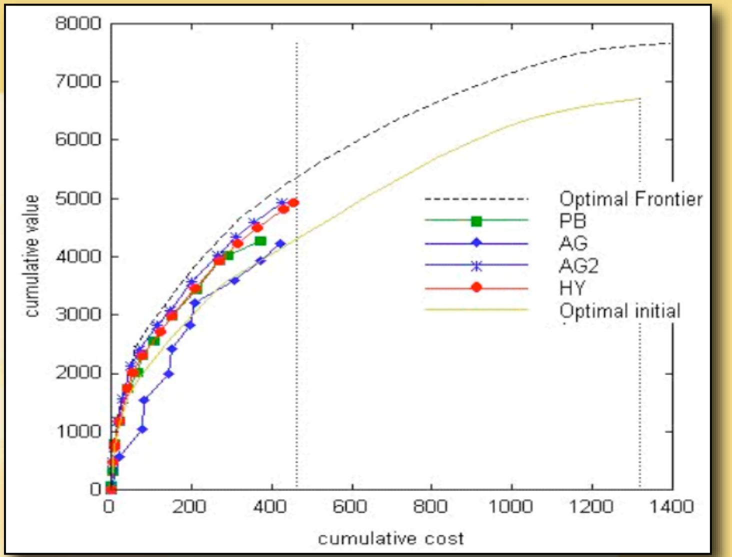
See reference 6, 11

- Initialization
  – R= Determine num_req
  – B = number "base requirements" (those known in iteration 1)
  – heap = {1,2,….,B, B+1, …R} sorted by (value/cost)
  – plan = heap

• PB is a one-time global search
• Ignores any changes due to value volatility

- Simulation:
  – Run down entire plan, left to right
  – Pause every value iteration to adjust requirements value

Performance measures



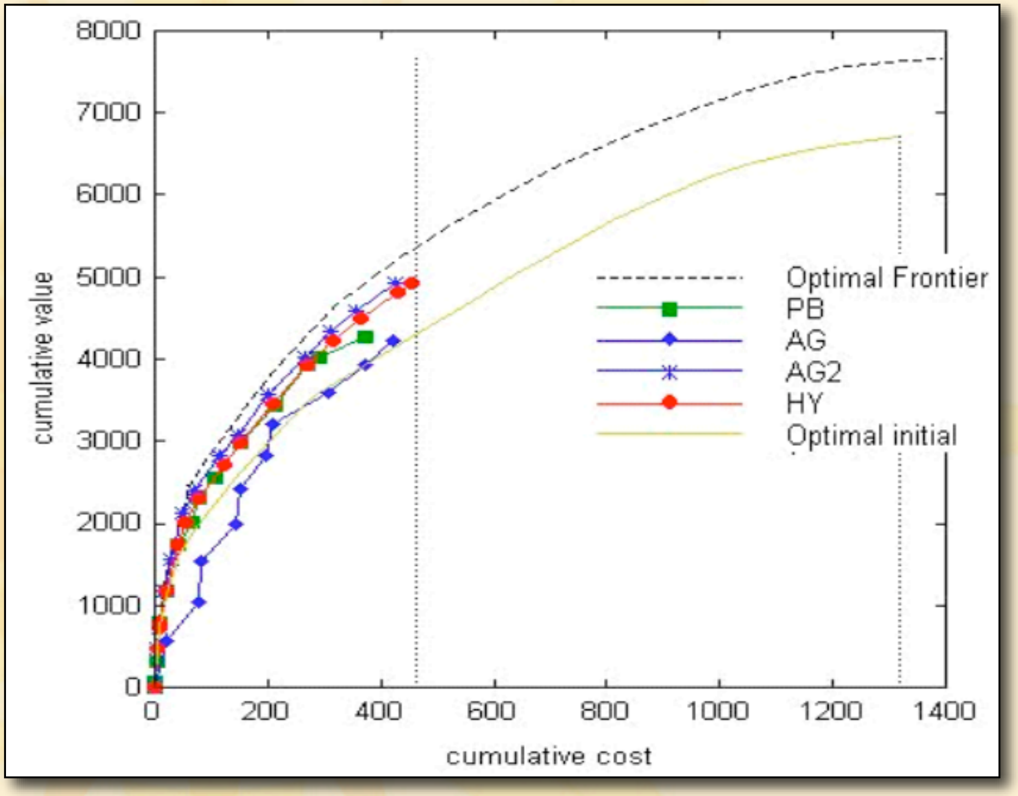
medium dynamism
$\lambda = 1.4$, $\sigma = 15\%$

- Control parameters
  - median new requirements discovered per iteration: $0.001 \le \lambda \le 20$
  - Requirements value volatility: $0.1\% \le \sigma \le 200\%$

- Cumulative
  - Of = Optimal frontier- "after the fact" of ordering of all requirements
    - Note: uses more information that available at any particular iteration
    - Represents maximum possible value.
  - Oi= Optimal initial: ordering the requirements using the initial values
  - Dynamism = Of - Oi (low if initial ordering is best requirements prioritization)

One trial results (1 of 3)

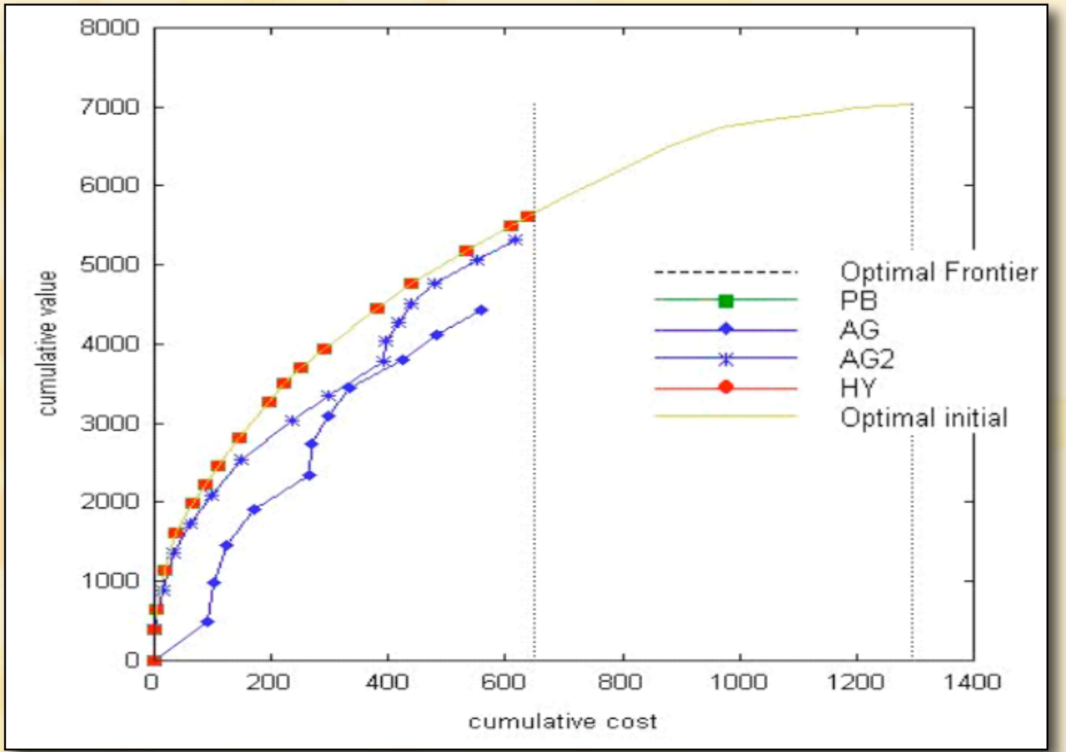
medium dynamism
$\lambda = 1.4, \sigma = 15\ \%$



- Extreme strategies (PB,AG) fail for this medium case.
- AG2 and HY perform best

One trial results (2 of 3)

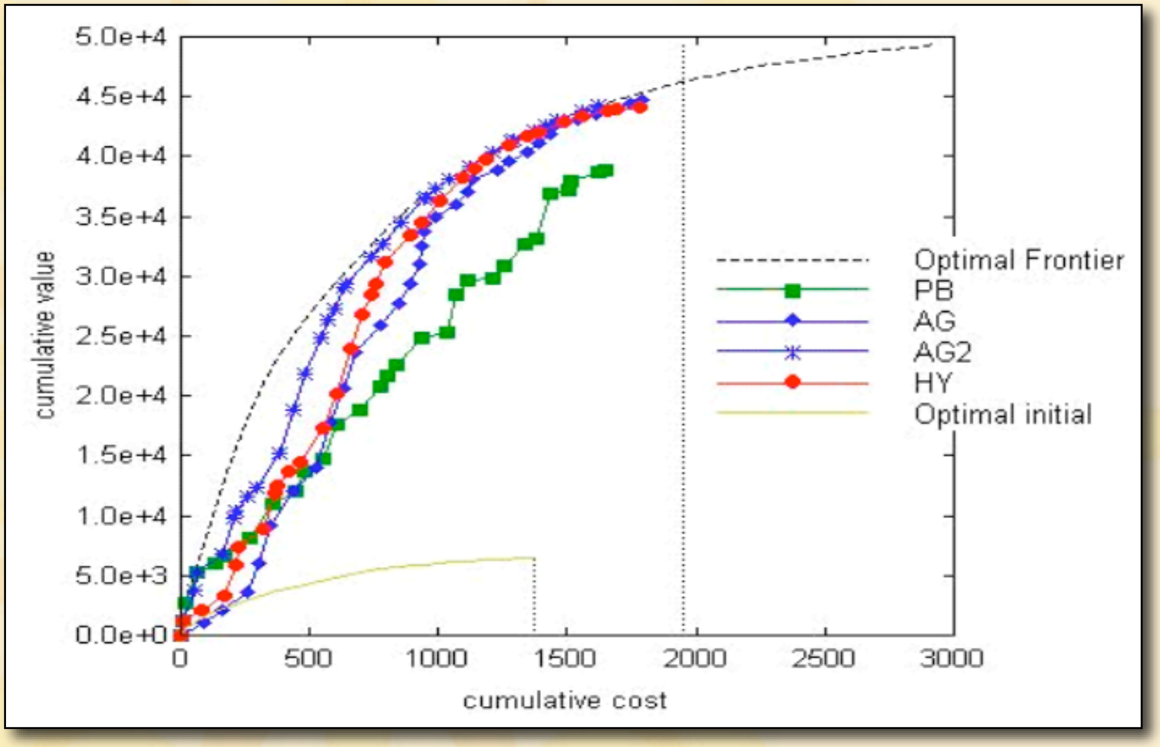
low dynamism
$\lambda = 0.001$, $\sigma = 0.1$ %



- Optimal initial = optimal frontier
- Expect: PB work best, AG worst
- Actual: HY/ PB best, both AGs worse
- And standard AG worst of all

One trial results (3 of 3)

high dynamism
$\lambda = 20, \sigma = 200$ %



- Expect: PB work worst, AG best
- Actual: PB worst, standard AG needs some help
- AG2 or HY beats AG

So there's more to life than standard AG

1000 trial results

- tb= total benefits
- tc = total costs
- Ben = benefit = tb - tc
- CB = tb/tc
- Int = integral= area under tb/tc curve
- FR = ratio of final to the optional frontier

TABLE IV. AVERAGE RANKS FOR N=1000 TRIALS

	LOW σ=0%	MED σ=15%	HIGH σ=200%
HIGH λ=20	*Value:* AG2 *Cost:* PB *Integral:* AG2 *Ben:* AG2 *CB:* AG2 *FR:* AG2	*Value:* AG2 *Cost:* PB *Integral:* AG2 *Ben:* AG2 *CB:* AG2 *FR:* AG2	*Value:* AG *Cost:* PB *Integral:* HY *Ben:* AG *CB:*HY, AG *FR:* HY
MED λ=1.4	*Value:* HY *Cost:* PB *Integral:* HY *Ben:* HY *CB:* HY *FR:* HY	*Value:* HY *Cost:* PB *Integral:* HY *Ben:* HY *CB:* HY *FR:* HY	*Value:* HY *Cost:* PB *Integral:* HY *Ben:* AG *CB:* HY *FR:* HY
LOW λ=0	*Value:* HY, PB *Cost:* PB *Integral:* HY *Ben:* HY, PB *CB:* PB *FR:* HY, PB	*Value:* HY *Cost:* PB, AG2 *Integral:* HY *Ben:* HY *CB:* HY *FR:* HY	*Value:* HY *Cost:* HY *Integral:* HY *Ben:* HY *CB:* HY *FR:* HY

HY: Hybrid, PB: Plan-based, AG: Agile, AG2: Agile cost-benefit

- HY dominates for Integral (7/9 experiments)
- PB dominates for cost (8/9 experiments)
- AG2 dominates for high λ and low to medium σ

Conclusion: Agile beats PB?

- That is the wrong question
- Better question(s)
  - What is the rate of new project requirements and value volatility?
  - What does the simulator say is the best combination of strategies for your domain?
- In these studies
  - No strong case for either PB or AG
  - (which may not hold for your next project)
- No more trite answers
  - Tune methods to local environments

not
rocket science

more studies
like this?

Challenge

- Is anyone surprised?
  - Hybrid combinations do better than the obsessive application of diametrically opposed extremes.
- How much of our time is spent debating needlessly polarized viewpoints?
  - plan vs agile
  - procedural vs object
  - model checking vs testing
  - etc
- Of course large diverse teams will combine methods
  - We should research those combinations
- More coalition
  - less opposition

Questions or Comments or … ?

Back up slides

Motivation

- Data drought
  - The COCOMO data ceiling
    - (1997,2008) records = (161,161)
  - NASA's data ceiling: 2005 - 2007 (+5)
- If we can't reason fully from data,
  - Reason mostly from models
  - Informed, minimally, by current records
- This work:
  - model-based reasoning on requirements prioritization strategies
  - Study humans like atoms in a crystal
    - Stochastic, but with stable emergent properties
  - We have (just) enough data + models to report and exploit regularities in the behavior of humans developers.
- Main result:
  - new prioritization halfway between two polarized positions
    - Not "agile is best"
    - Not "pre-planning is best"
    - But a new hybrid strategy

The “Separation of Concerns” legacy

- “The notion of ‘user’ cannot be precisely defined, and therefore has no place in CS or SE.”
  - Edsger Dijkstra, ICSE 4, 1979
- “Analysis and allocation of the system requirements is not the responsibility of SE, but a prerequisite for their work.”
  - Mark Paulkat al., SEI Software CMM v.1.1, 1993
- Now, after decades of SE…
  - No more separation?
  - Study humans like atoms in a crystal
    - Stochastic, but with stable emergent properties
  - We have (just) enough data + models to report and exploit regularities in the behavior of humans developers.

Cao, L., Ramesh, B., Requirements Engineering Practices: An Empirical Study, IEEE Software, Vol 25, p60- 67, 2008
- Data from 16 companies