Real world effort estimation…

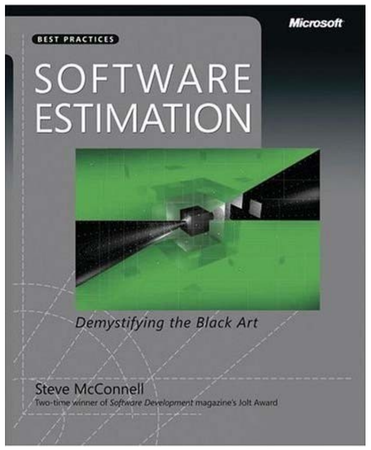
I'M GETTING WILDLY DIFFERENT ESTIMATES FOR HOW LONG IT WILL TAKE TO WRITE THE SOFTWARE.
BASED ON MY EXPERIENCE, I SAY TAKE THE LONGEST ESTIMATE AND MULTIPLY IT BY THREE.
IS EXPERIENCE EXACTLY THE SAME AS PESSIMISM?
EXPERIENCE IS MUCH WORSE.
Dilbert.com @ScottAdamsSays
10-20-16 © 2016 Scott Adams, Inc. /Dist. by Universal Uclick

No More Guessing

Effort Estimation with Jupyter Notebook

Sebastian Kübeck
Mastodon, LinkedIn, Twitter: @skuebeck
sebastian.kuebeck@aon.at
https://sebastiankuebeck.wordpress.com/

Terminology


BEST PRACTICES
Microsoft
SOFTWARE ESTIMATION
Demystifying the Black Art
Steve McConnell
Two-time winner of Software Development magazine's Jolt Award

An **estimate** is an approximation based on incomplete information.

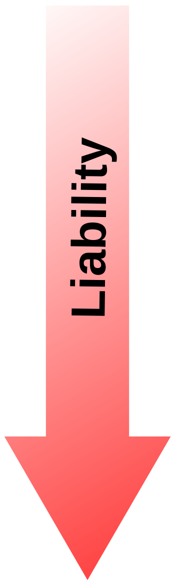
An **ambition** is the commitment to do better than the estimate, if possible.

A **target** is a statement of a desirable business objective. E.g.

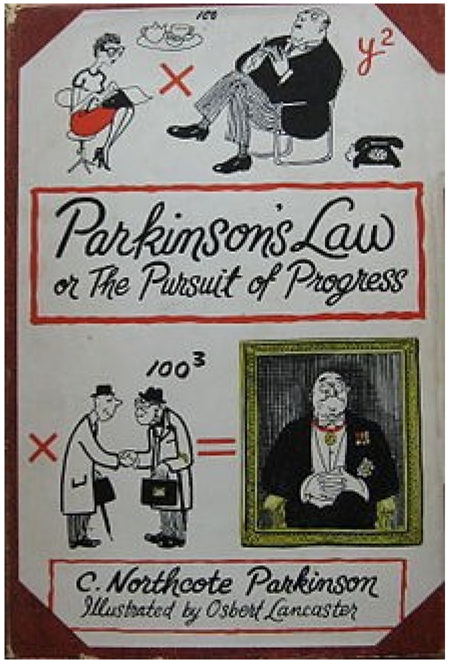
- "We need to have Version 2.1 ready to demonstrate at a trade show in May."
- "We need to have this release stabilized in time for the holiday sales cycle."

A **commitment** is a promise to deliver defined functionality at a specific level of quality by a certain date.

An **offer** is a conditional promise which becomes a binding contract if the offeree accepts it.



Parkinsons Law is an urban legend!



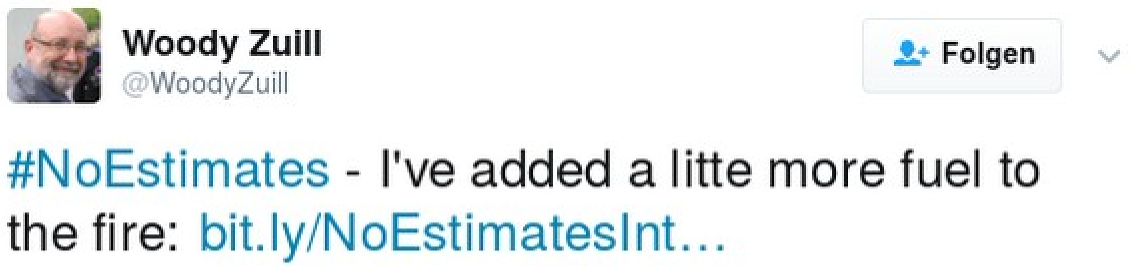
Originally, Parkinson's law is the adage that "**work expands so as to fill the time available for its completion**", and the title of a book which made it well-known.

However, in current understanding, Parkinson's law is a reference to the **self-satisfying uncontrolled growth of the bureaucratic apparatus** in an organization.

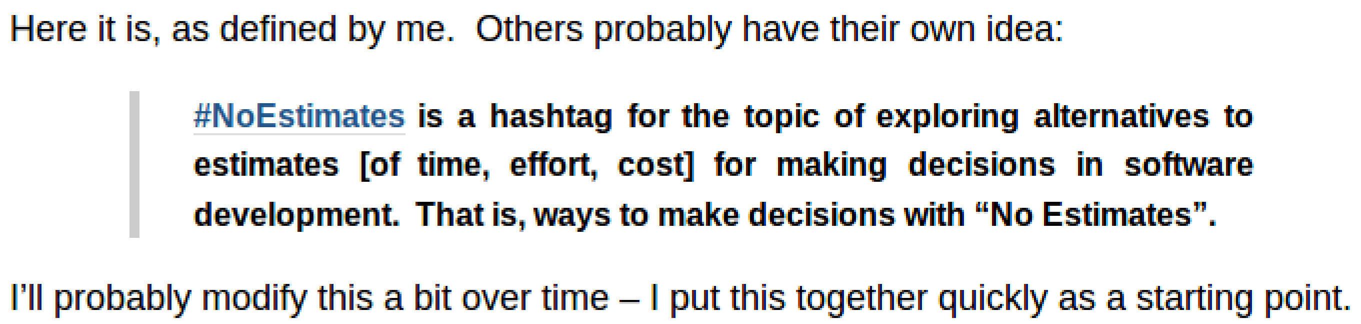
https://en.wikipedia.org/wiki/Parkinson's_law

Why not drop estimation at all?

2012: First use of #NoEstimates ....



#NoEstimates - I've added a litte more fuel to the fire: bit.ly/NoEstimatesInt…

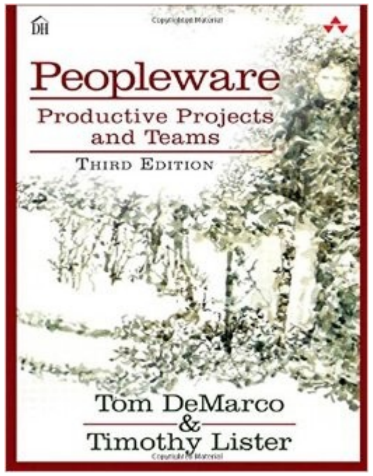


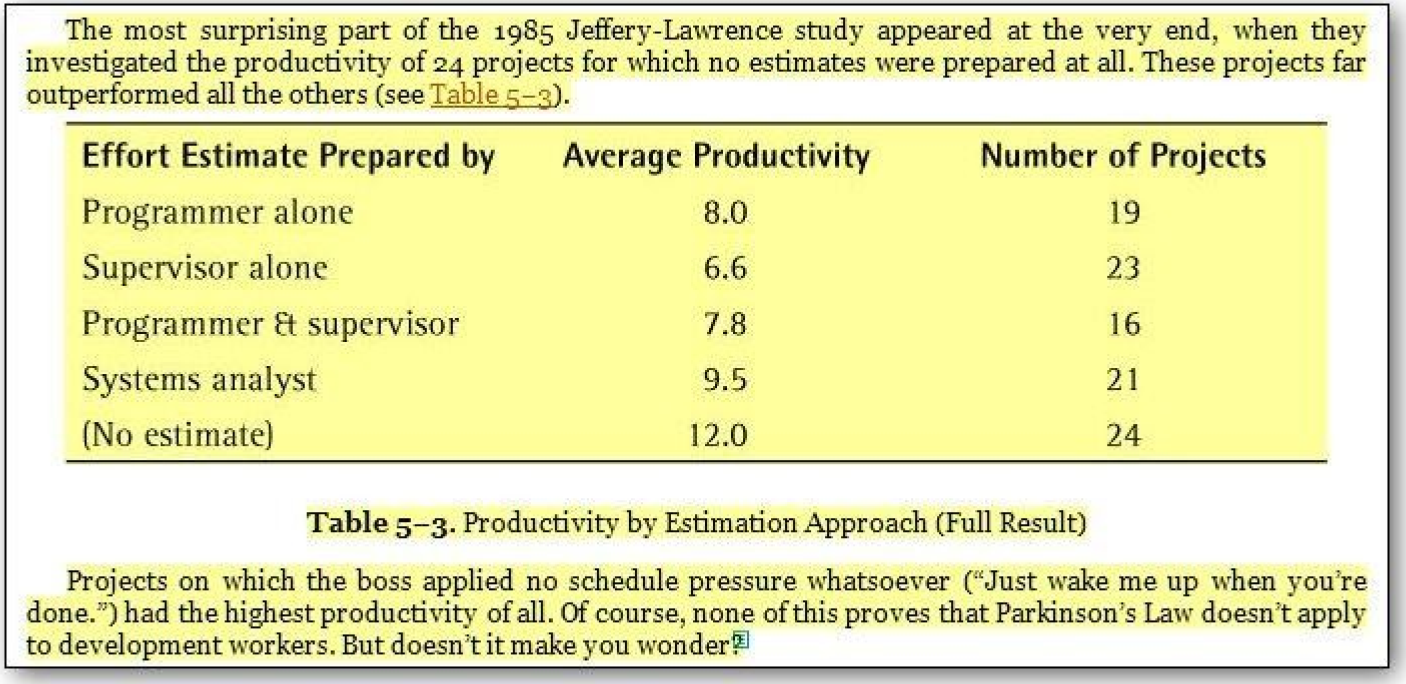
Here it is, as defined by me. Others probably have their own idea:

> **#NoEstimates is a hashtag for the topic of exploring alternatives to estimates [of time, effort, cost] for making decisions in software development. That is, ways to make decisions with "No Estimates".**

I'll probably modify this a bit over time – I put this together quickly as a starting point.

Studies: Teams that don't estimate are more productive!





The most surprising part of the 1985 Jeffery-Lawrence study appeared at the very end, when they investigated the productivity of 24 projects for which no estimates were prepared at all. These projects far outperformed all the others (see Table 5–3).

Effort Estimate Prepared by	Average Productivity	Number of Projects
Programmer alone	8.0	19
Supervisor alone	6.6	23
Programmer & supervisor	7.8	16
Systems analyst	9.5	21
(No estimate)	12.0	24

Table 5–3. Productivity by Estimation Approach (Full Result)

Projects on which the boss applied no schedule pressure whatsoever ("Just wake me up when you're done.") had the highest productivity of all. Of course, none of this proves that Parkinson's Law doesn't apply to development workers. But doesn't it make you wonder?[2]

#NoEstimates today

2023: #NoEstimates is dead!

Reasons:

1. Proponents could not agree on definitions and methodology.
2. Truism: It is not possible to think about the future in a reasonable way without any form of estimation.
3. It turned out that problems with estimates have the following reasons and they can not be solved by not estimating!
    - Lack of knowledge/education
    - Bad management practices

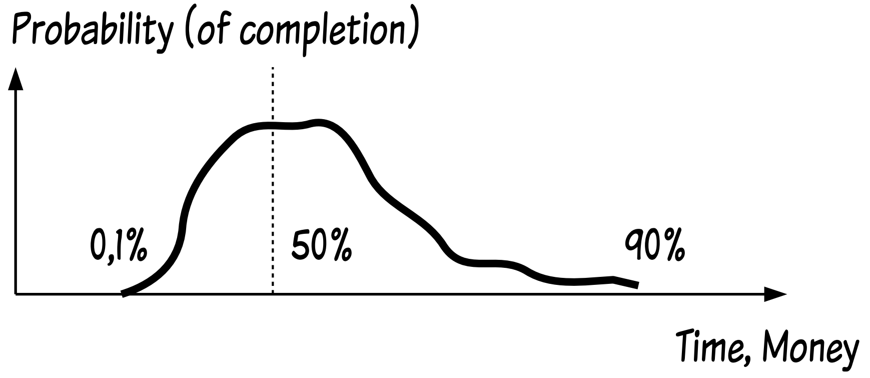
#NoEstimates Spin-Off

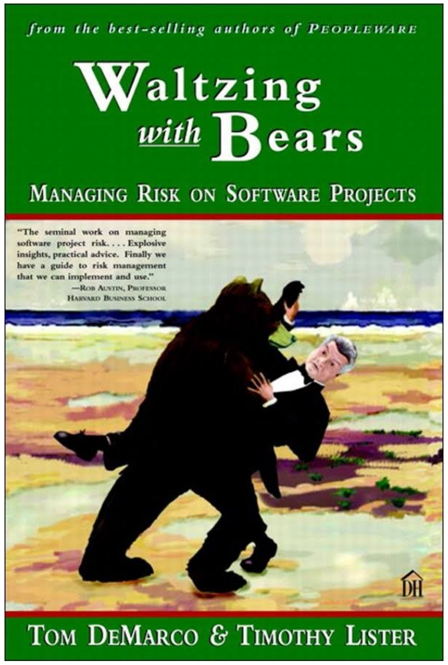
Estimations based on previous Data!

- Estimation is done by the computer, based on previous project data.

Estimation based on Previous Data

Goal: Figure out tasks, estimating task size, summing up tasks, come up with a probability distribution instead of a number.




from the best-selling authors of PEOPLEWARE
Waltzing with Bears
Managing Risk on Software Projects
"The seminal work on managing software project risk. . . . Explosive insights, practical advice. Finally we have a guide to risk management that we can implement and use."
—Rob Austin, Professor Harvard Business School
Tom DeMarco & Timothy Lister

Estimation from previous Project Data

Questions

- How to I get to an offer for a fixed priced project?
- How much money can I make with this projects? What are the financial risks?
- When will the project be finished?

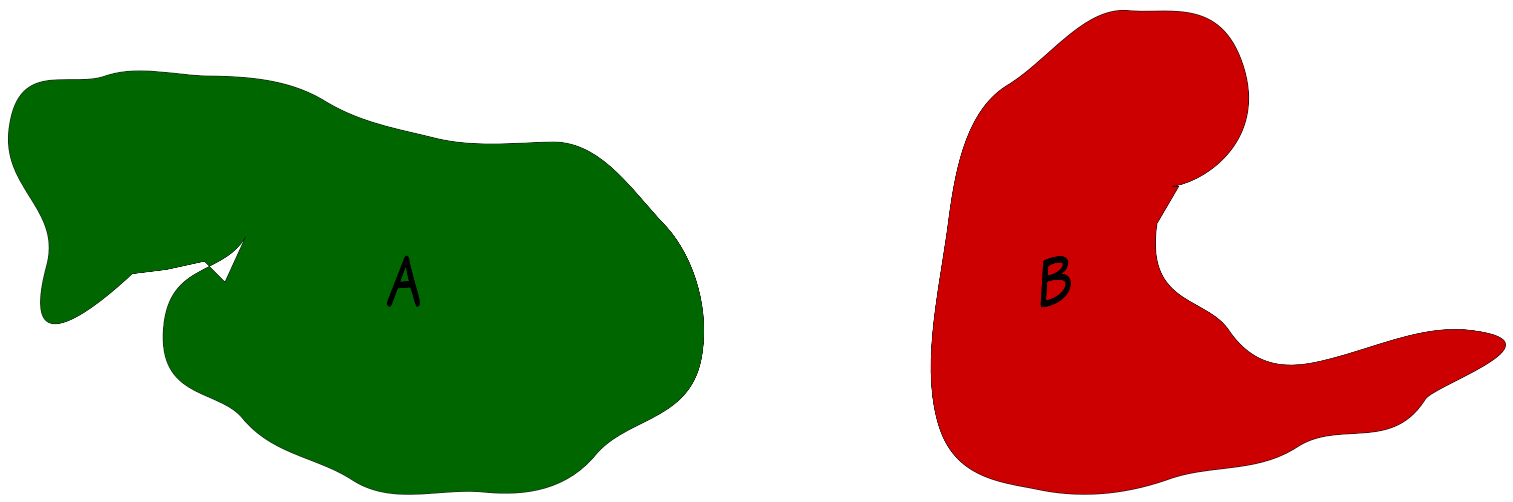
Estimation from previous Project Data

Environment

- **Scrum** is used as development process.
- Development is orgenaized in Teams with 4-8 people.
- Work is done within fixed time Intervals called **Sprints**.
- Work is split in **Stories** as unit of Work.
- **Stories** are estimated in **Story Points**, a relative metric that’s specific to a team.
- **Velocity** is defined as **Story Points** completed per **Sprint**.

https://www.scrum.org/resources/scrum-guide

Story Points

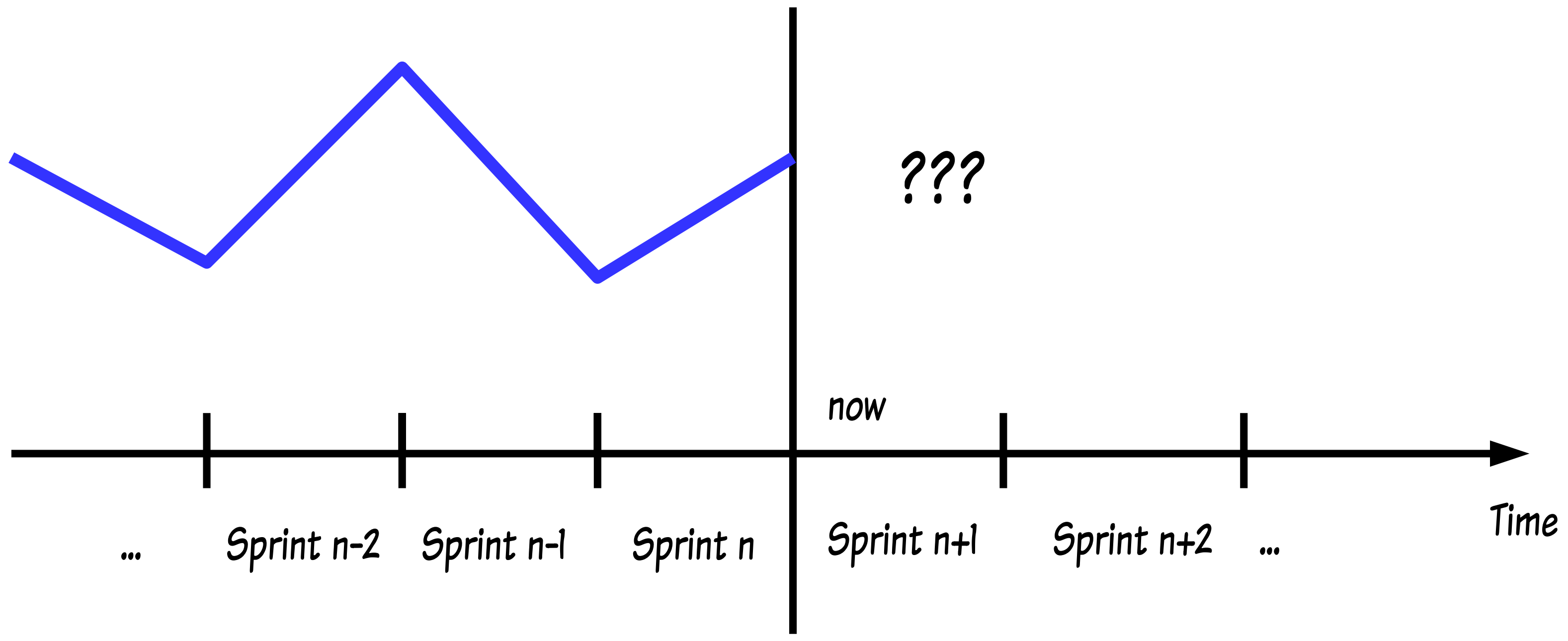


Flächeninhalt A:B ≈ 3:4

...in Story Points, this is 4:3, 8:6 or 40:30, every team usees their own story points

Problem: The velocity is not constant!

Velocity = Story Points/Sprint



Example: Fixed Price Project

Given:

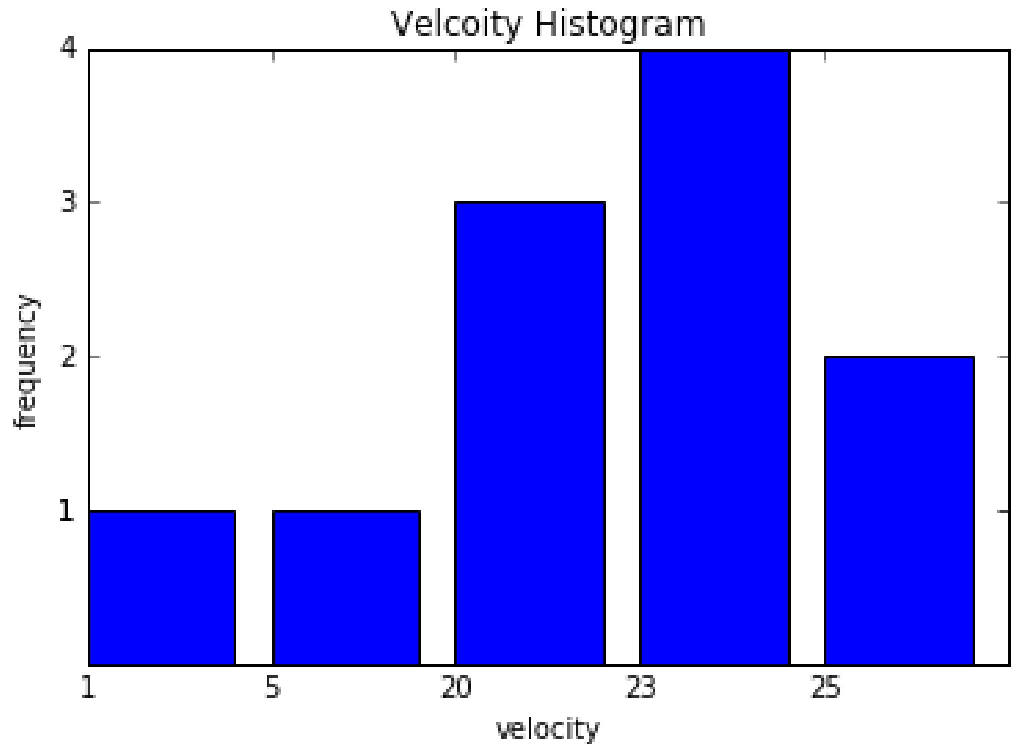
- Projekt requires **50 Story Points**
- The velocity distribution is the same as in the past

To be determined:

- **Best Case**: How many Sprints does it take to complete the project in an optimal situation? How likely ist this case?
- **Bad Case:** How many sprints do I have to sell, not to make a loss with a probability of 95%?
- **Worst Case:** What is the maximum number of Sprints required to complete the projcet? How likely ist this case?
- **Expected Gain:** How many Sprints do I have to sell in order to make a profit?

Velocity

Assumption: Velocity distribution will not change in the future



Edge Cases from the Histogram

Best Case

- Maximum Velocity: 25 Story Points
- Number of Sprints for maximum Velocity: 50/25 = **2 Sprints**
- Probability:

  (Frequency 25 Story Points/Number of Measurements)$^{\text{Sprints}}$

  result: $(2/11)^2$ = **3,31%**

Worst Case:

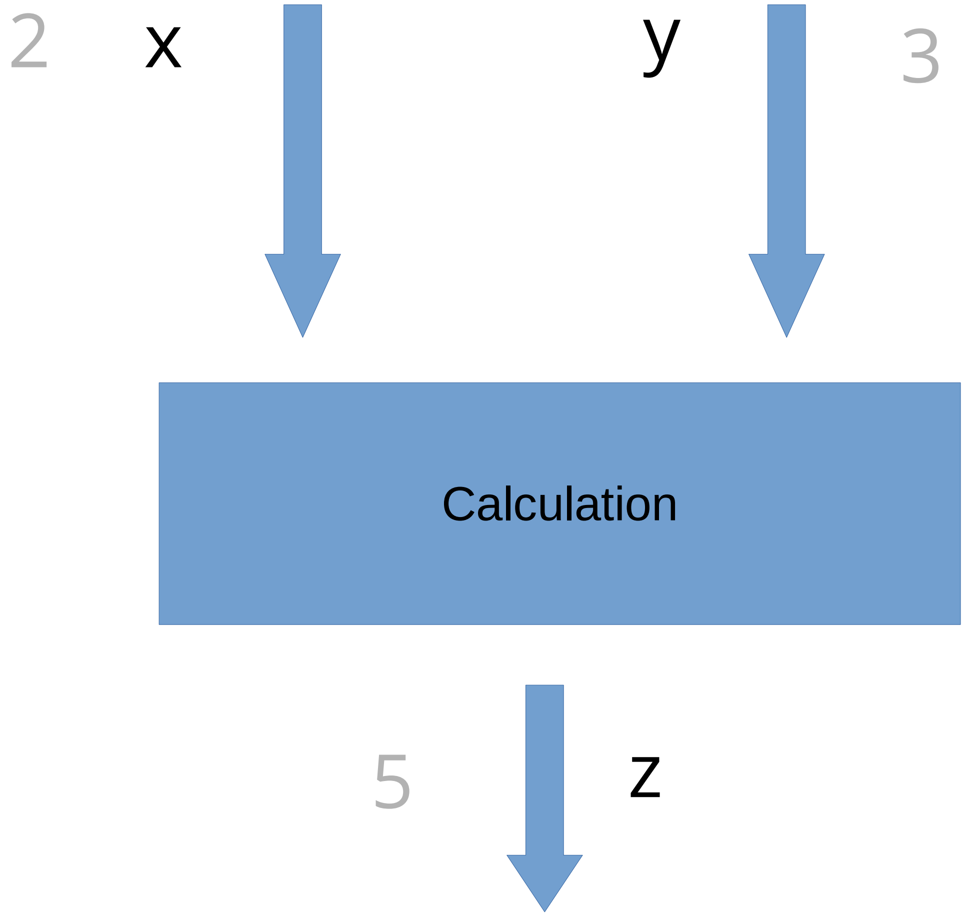
- Minimum Velocity: 1 Story Point
- Number of Sprints at minumum Velocity: 50/1 = 50 **Sprints**
- Probability:

  (Probability 1 Story Points/Number of Measurements)$^{\text{Sprints}}$

  result: $(1/11)^{50}$ = **$8{,}5 \times 10^{-51}$**

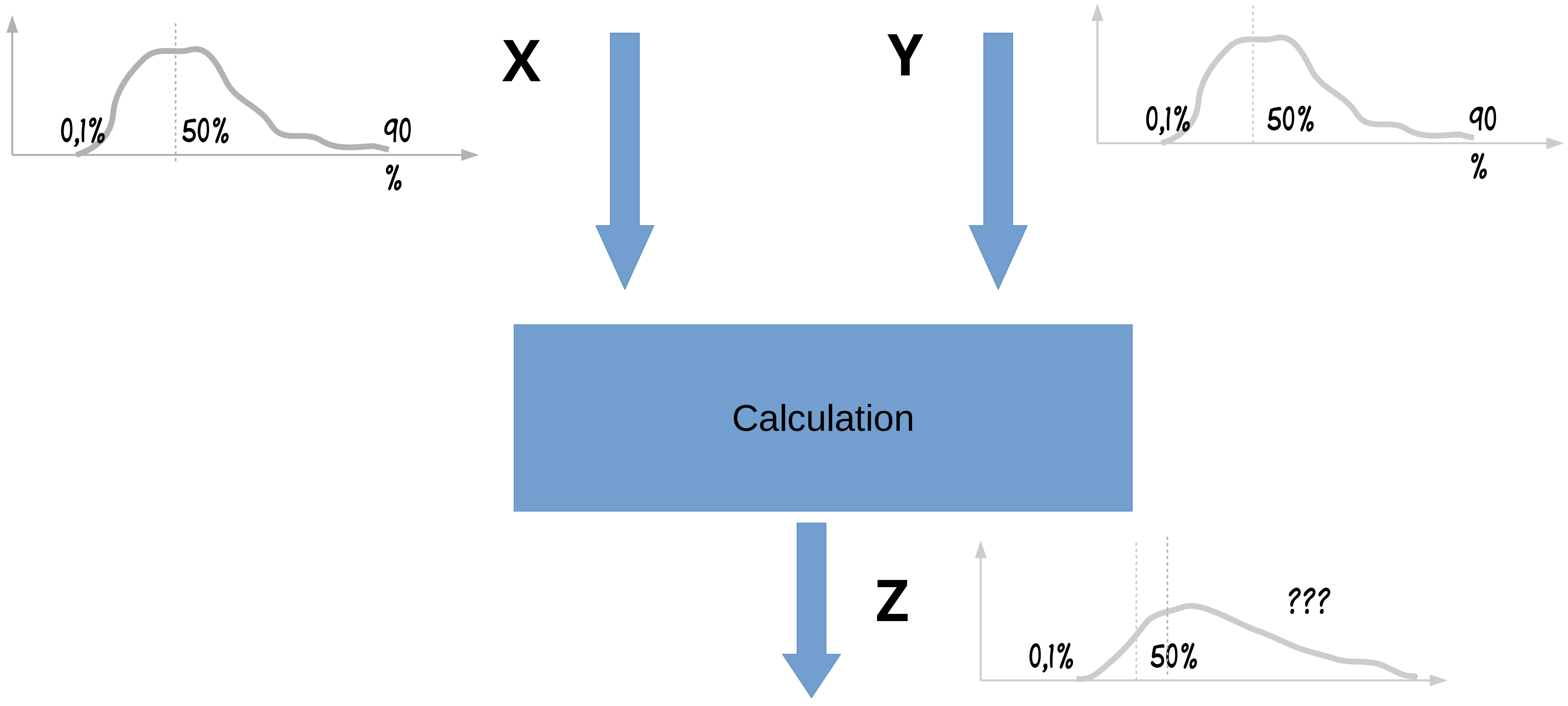
  ..way too improbable!

Deterministic System



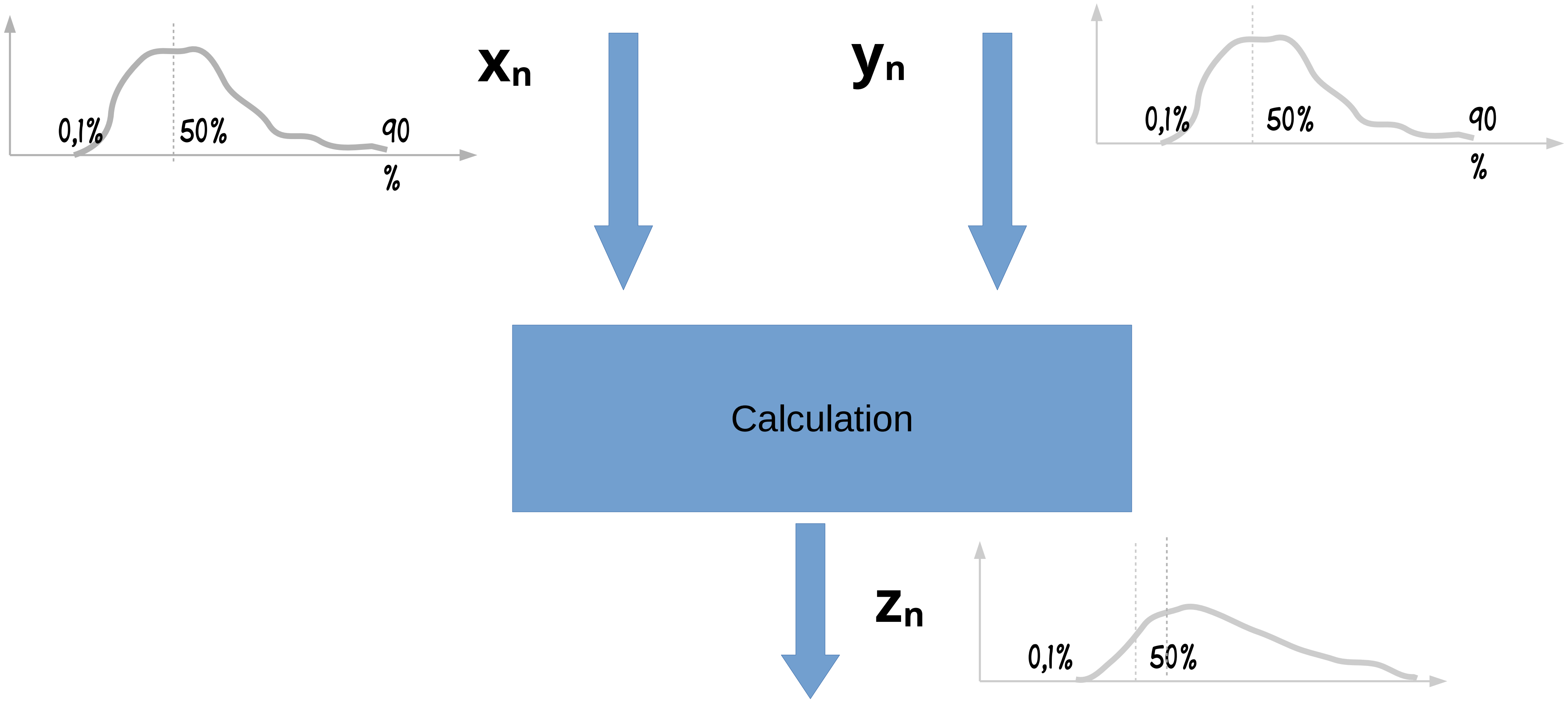
Task: x, y known, calculate z.

Probabilistic System



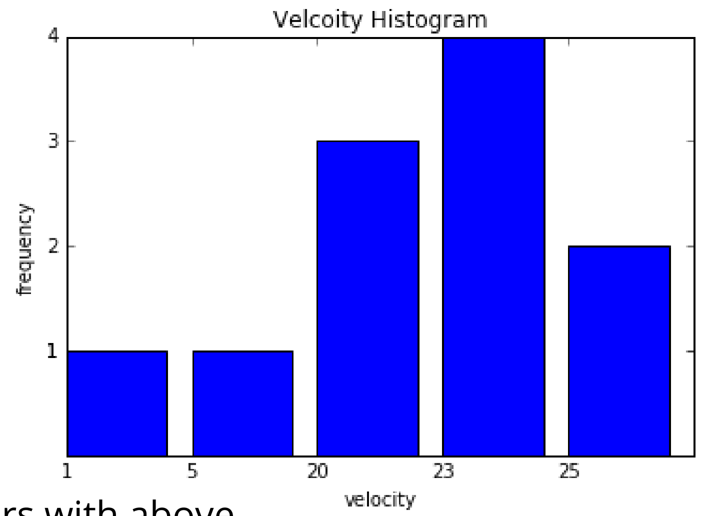
X,Y and Z are *Random Variables*!
Task: *distribution* of X, Y known, calculate *distribution* of Z.

Monte Carlo Approximation



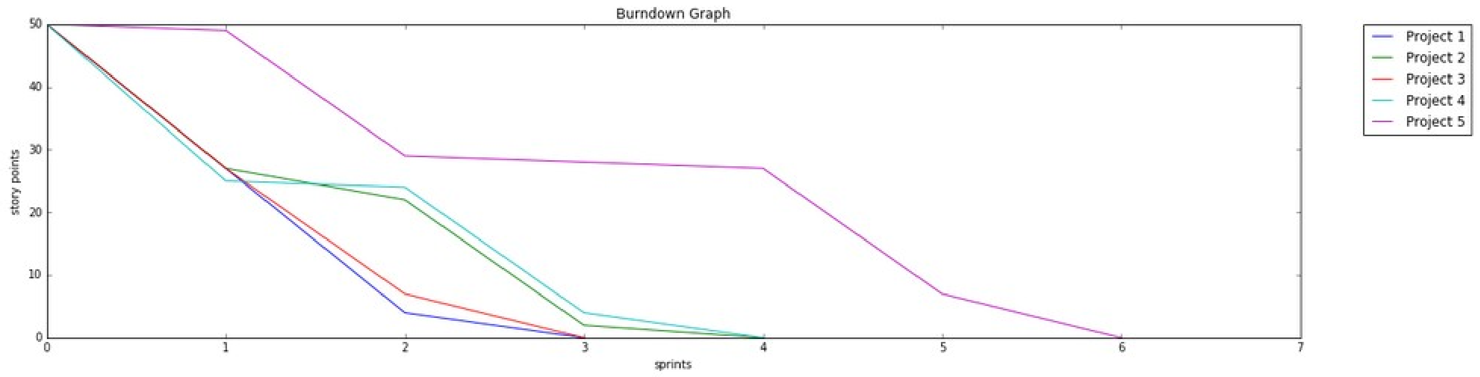
1. **Take n random samples x_n, y_n from Distributions X and y.**
2. **Calculate z_n from x_n, y_n.**
3. **The histogram of z_n will approximate Distribution of Z.**

Monte Carlo Method for Project outcomes



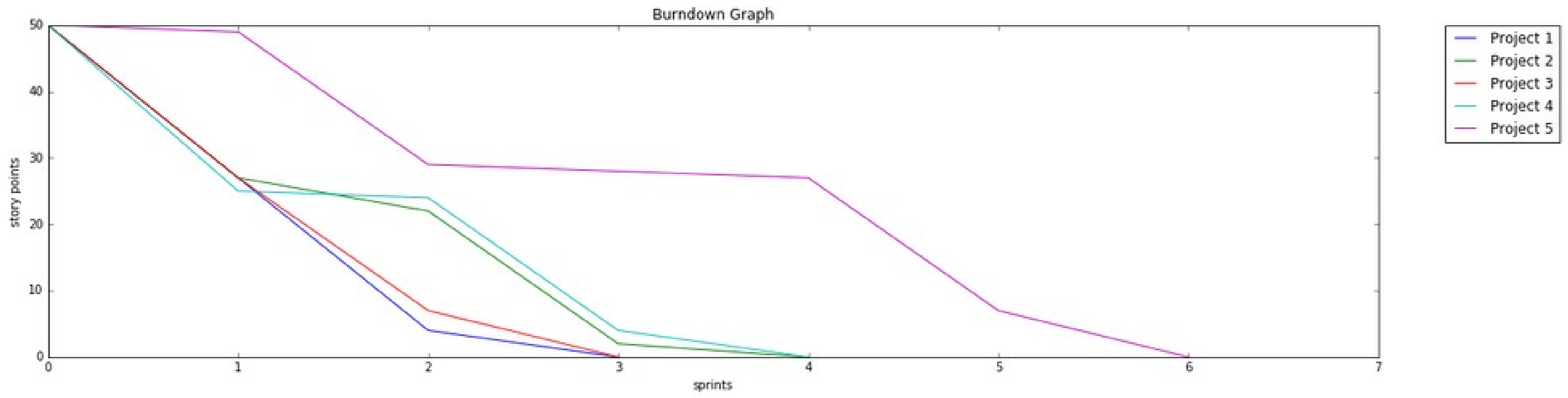
Random numbers with above distribution
= simulated Velocity = Story Points/Sprint

Sum up simulated Velocities until 50 Story Points are reached. Note the numner of Sprints



Burn-Down Graph

Burn-Down Graph for 5 possible project outcomes...



Monte Carlo Method

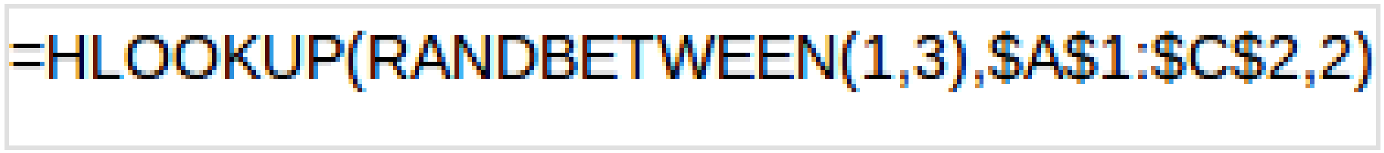
Generate random number with given distribution

Numpy (Python):

```
numpy.random.choice(a=Values, p=Propabilities)
```

https://docs.scipy.org/doc/numpy-dev/reference/generated/numpy.random.choice.html

Excel:

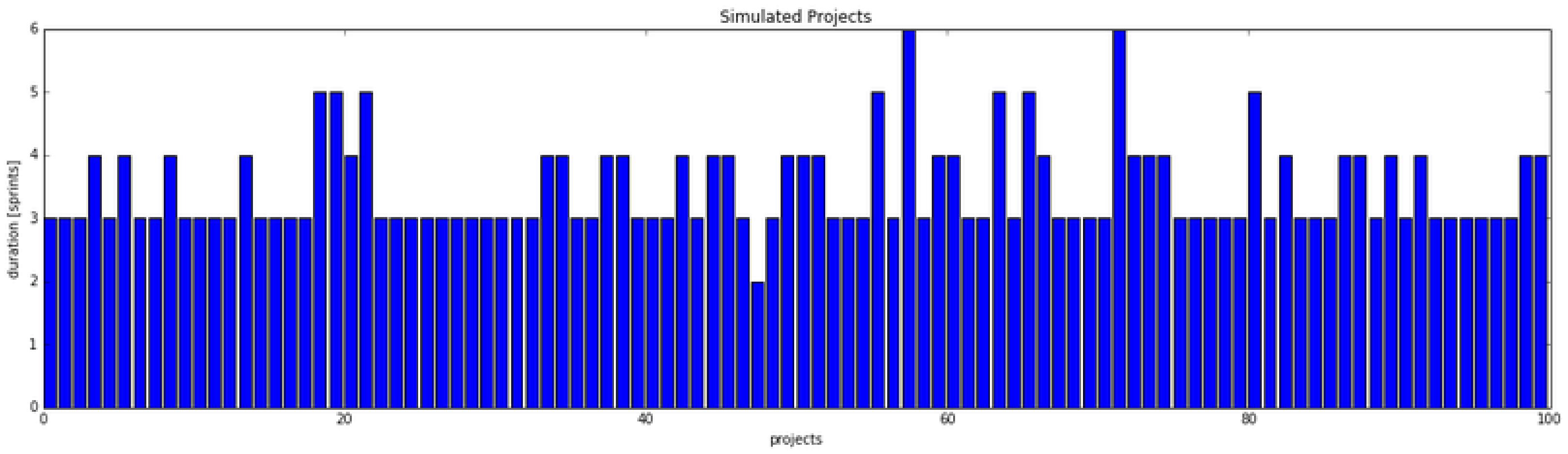


```
=HLOOKUP(RANDBETWEEN(1,3),$A$1:$C$2,2)
```

Michael de la Maza: Monte Carlo Planning Improves Decision Making, InfoQ 2017:
https://www.infoq.com/articles/monte-carlo-planning

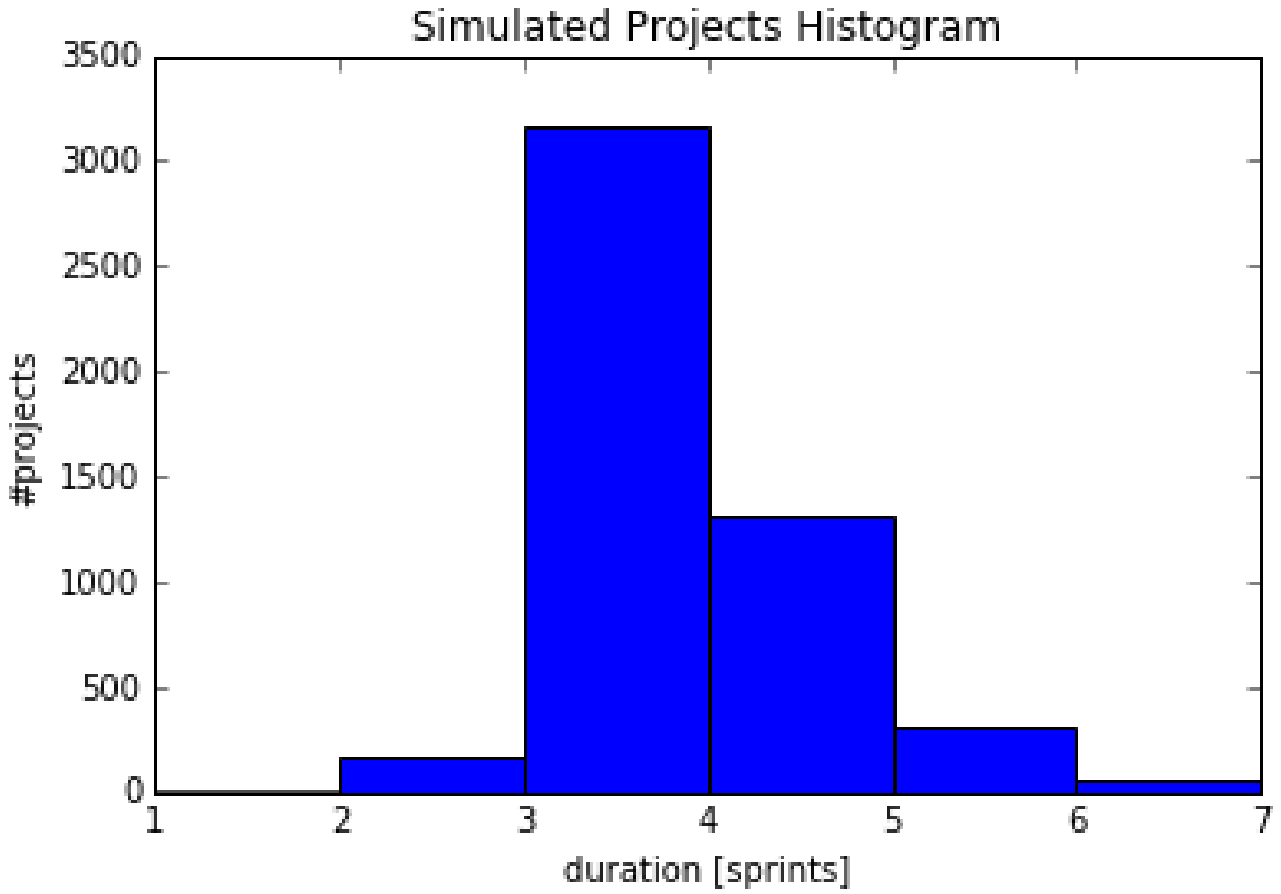
Number of Sprints required

Number of required Sprints for 1000 Project runs…

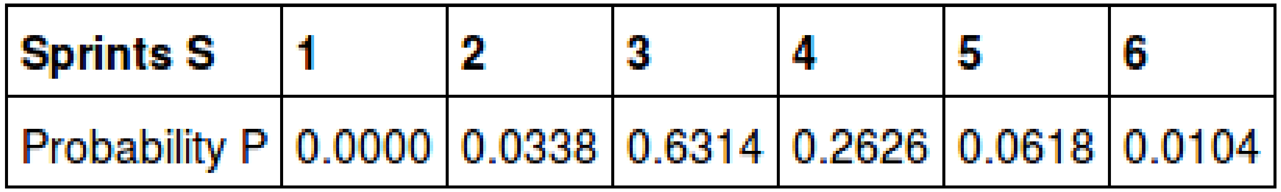


Histogram of Simulated Projects

5000 simulated projects later....



Histogram Evaluation



Sprints S	1	2	3	4	5	6
Probability P	0.0000	0.0338	0.6314	0.2626	0.0618	0.0104

Average Case:

We make money starting at 50%, that means we have to sell at least **3 Sprints**!

We could have calculated this immediately:

Average Velocity: 18.91

Average number of Sprintsl:

Story Points total/average Velocity

result: 50/18.91 = **2.64, rounded up yields 3 Sprints!**

Bad Case:

- Probability > 95% corresponds to 1 – 0.05%
- From the histogramm: **6 Sprints** (Probability: 99.38%)
- Expected gain: **6 – 2.64 = 3.36 Sprints**

Conclusion

Best Case:

- We need at least **2 Sprints** to finish the project!
- However, the probability is pretty low 3.31%!

Average Case:

- We need to sell at least **3 Sprints** to make money!

Bad Case:

- We need at least **6 Sprints** to be sure that we **don’t** loose money with a probability of 95%!

Worst Case:

- Only if we sell 50 Sprints we can be 100% certain that we don’t loose money.
- However, the probability that we actually need 50 Sprint is extremely unlikely.

Further Reading

Magically Predictable Software Production for Projects

Ralf Westphal, 2019

https://ralfw.de/magically-predictable-software-production-for-projects/Monte Carlo

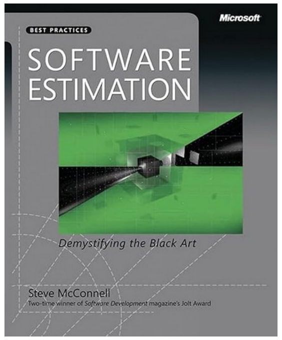
Estimation for Kanban Projects

Software Estimation: Demystifying the Black Art

Steve McConnell, 2009

https://leventov.medium.com/excerpts-from-software-estimation-demystifying-the-black-art-9cf80e2b9977

Comprehensive introduction to Software Estimation



BEST PRACTICES

Microsoft

SOFTWARE ESTIMATION

Demystifying the Black Art

Steve McConnell

Two-time winner of Software Development magazine's Jolt Award

Forecasting and Simulating Software Development Projects

Troy Magennis 2011

https://github.com/FocusedObjective/FocusedObjective.Resources/blob/master/README.md

Forecasting and Simulating Software Development Projects

Effective Modeling of Kanban & Scrum Projects using Monte-carlo Simulation

Troy Magennis

Planning without prior Data

How do I deal with the following situation?

- New team or major change.
- R&D Project.
- No idea if it can be accomplished.
- Not familiar with domain, technology

Planning without prior Data

Solution:

Don’t estimate **Effort** but **Value**!

Planning without prior Data

Process (yet unnamed)

1. Define **goal**
2. Figure out how much you are willing to invest to reach this **goal**.
3. Set up a **budget** based on defined investment.
4. Start working until...
   - **goal** is reached,
   - it is obvious that the goal cannot be reached,
   - The **budget** is used up.
5. Continue with 1

How to Estimate Value

Gilb Method

1. STAKEHOLDERS: Identify your most critical stakeholders.

2. OBJECTIVES: Identify the smart levels of their most critical value improvements.

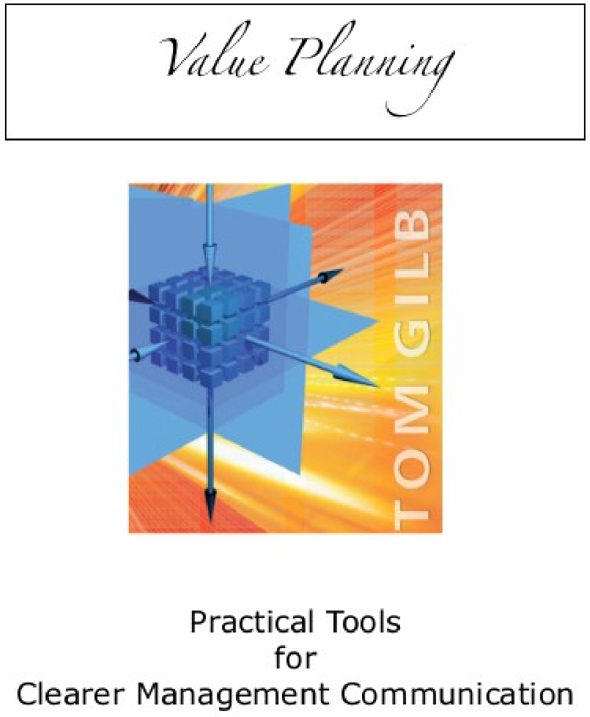
3. STRATEGIES: Identify potential strategies for delivering planned value levels to

stakeholders, at lowest cost and risk.

4. SMALL STEPS: Decompose strategies into suitably smaller deliverable increments.

5. DELIVER VALUE: Attempt to deliver measurable value to some stakeholders.

6. LEARN: Measure results and costs; then decide if you are on track, or need to change something. Continue the process until all goals reached.



Value Planning

TOM GILB

Practical Tools
for
Clearer Management Communication

https://www.gilb.com/store/2W2zCX6z