Project Estimation

Bob Raman, Aug 2015

Acknowledgments

- Michael Galloway Mclean for proof reading and providing some great feedback.

Contents

Why Estimate?

- We have fixed amount of monies so we need to decide whether we do project A or B?
- We need to plan dependencies between front-end and back-end teams.
- Add more features to an existing project.

What does business want?

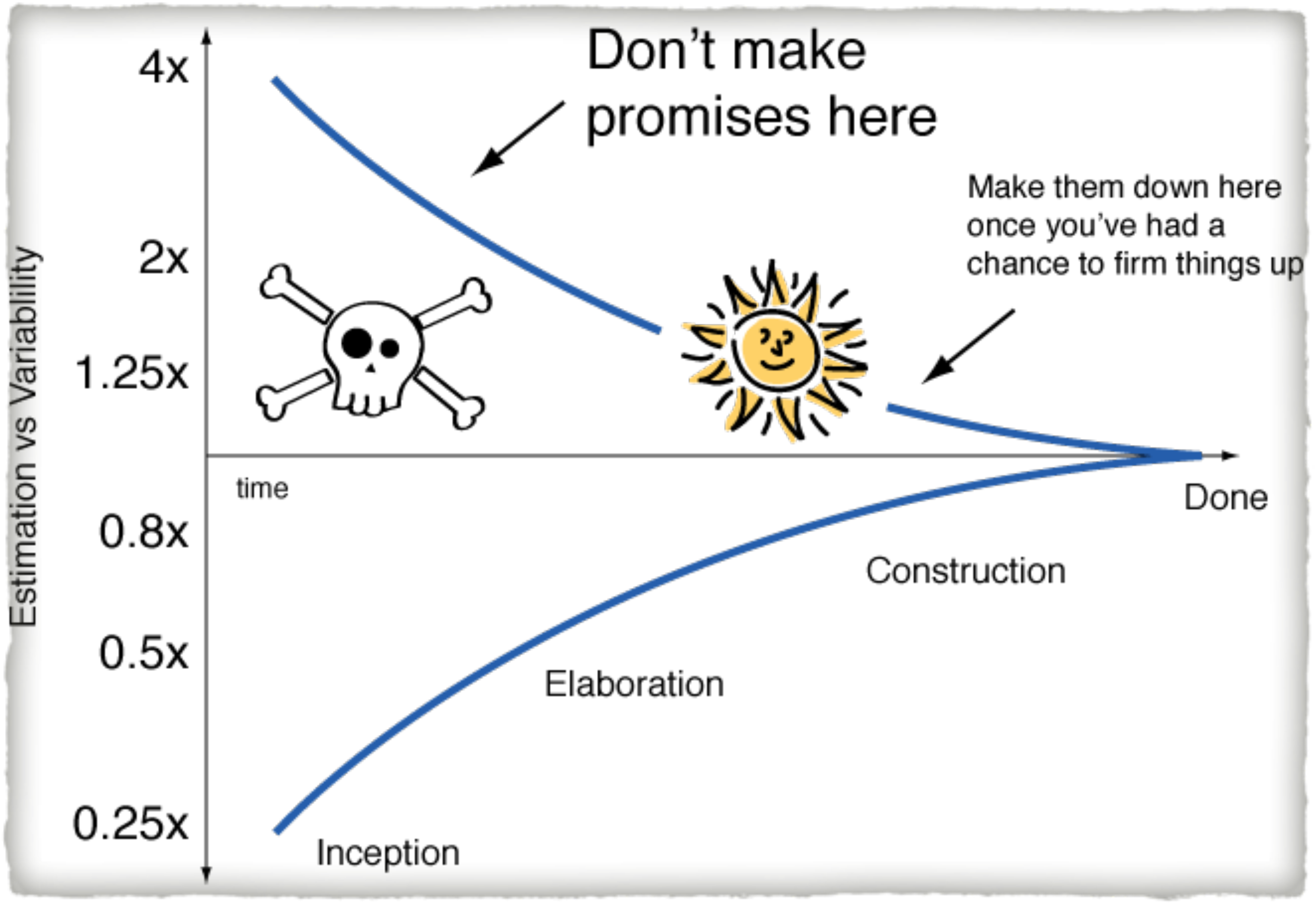
- *How much is this work going to cost me?*
- *When will we deliver it to the customer?*

- Estimate *effort* - not complexity
  - Start with **Relative Effort**.

Currency?

- *"A story-point estimate is an amalgamation of the amount of effort involved in developing the feature, the complexity of developing it, the risk inherent in it."* (Mike Cohn, 2010 [2])
- Also take account of volume of work

Cone of uncertainty



http://www.agilenutshell.com/cone_of_uncertainty

Accurate not Precise!

- Development is *innovation* not *construction*!
- Estimates are educated guesses!
- What accuracy are we after?
  - Order of magnitude ok?
  - Optimistic;Likely;Pessimistic
  - 90% confidence
- Accuracy has a cost!

Pre-requisites

- Ranked user story backlog
- Team who does the work does the estimates
- User Stories are not too large
- User Stories have enough detail to estimate
- Architecture/technology has been worked out.

Done-Done

- Scrum advocates that you are "Done-Done" at the end of each iteration.
  - Critical with release at a cadence

Process

- Pick a story that feels smallish (reference)
- Relatively estimate stories against the reference story
- Put each story into a bucket - 1,2,3,5,8,13
- Cap your stories to “n” points. i.e. do not play them unless below cap.
  - Ideally n==5
- Triangulation - Review the stories in each bucket to see if the size makes sense.

Exceptional Cases: Multiple Streams

- Initial joint planning poker session together to help dev understand the baseline for Story Points.

Who does the estimation?

- A story describes a business need. This is generally expressed in terms of a stakeholder experience.

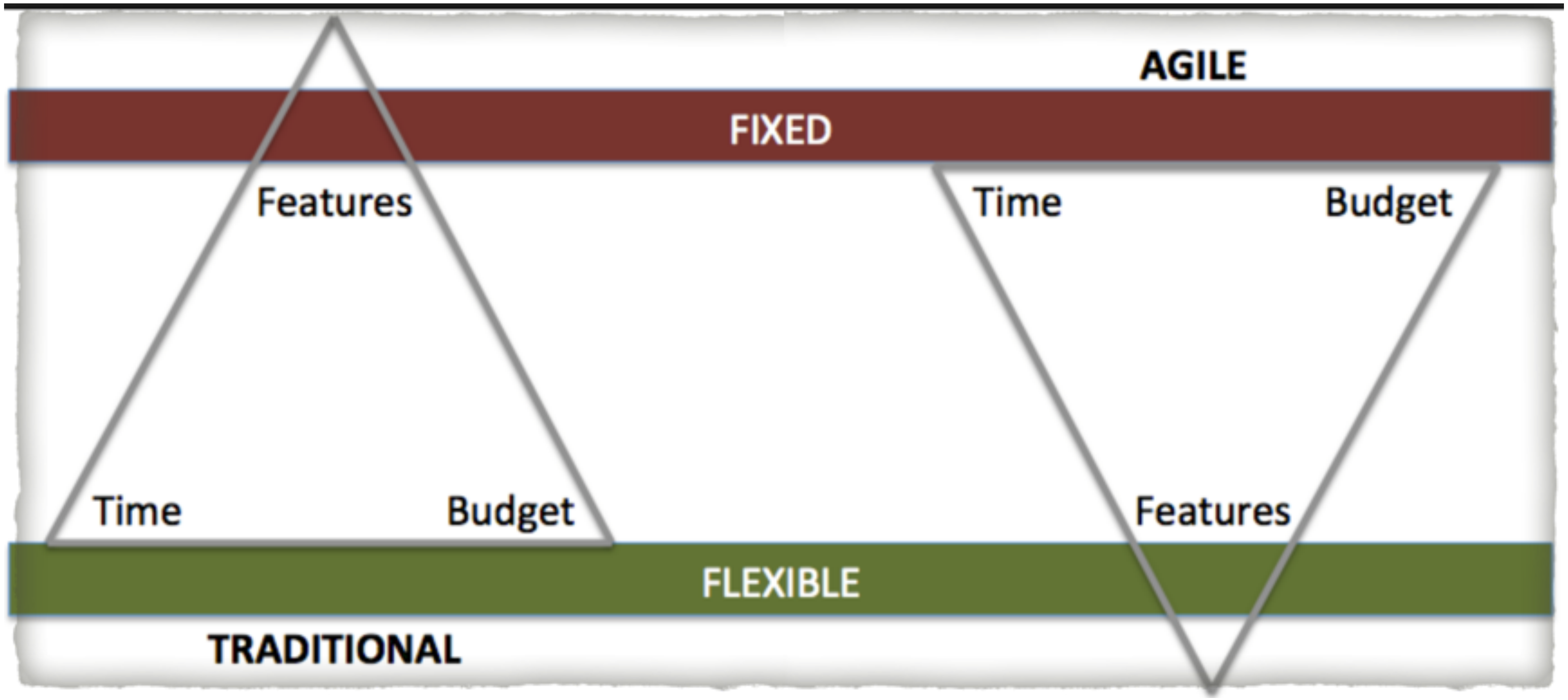
- Story points should represent what it takes to get a story into *"Done"* state.

- Story points are owned by the team, not the developers. Testers should be part of the estimation to help developers give better estimates.

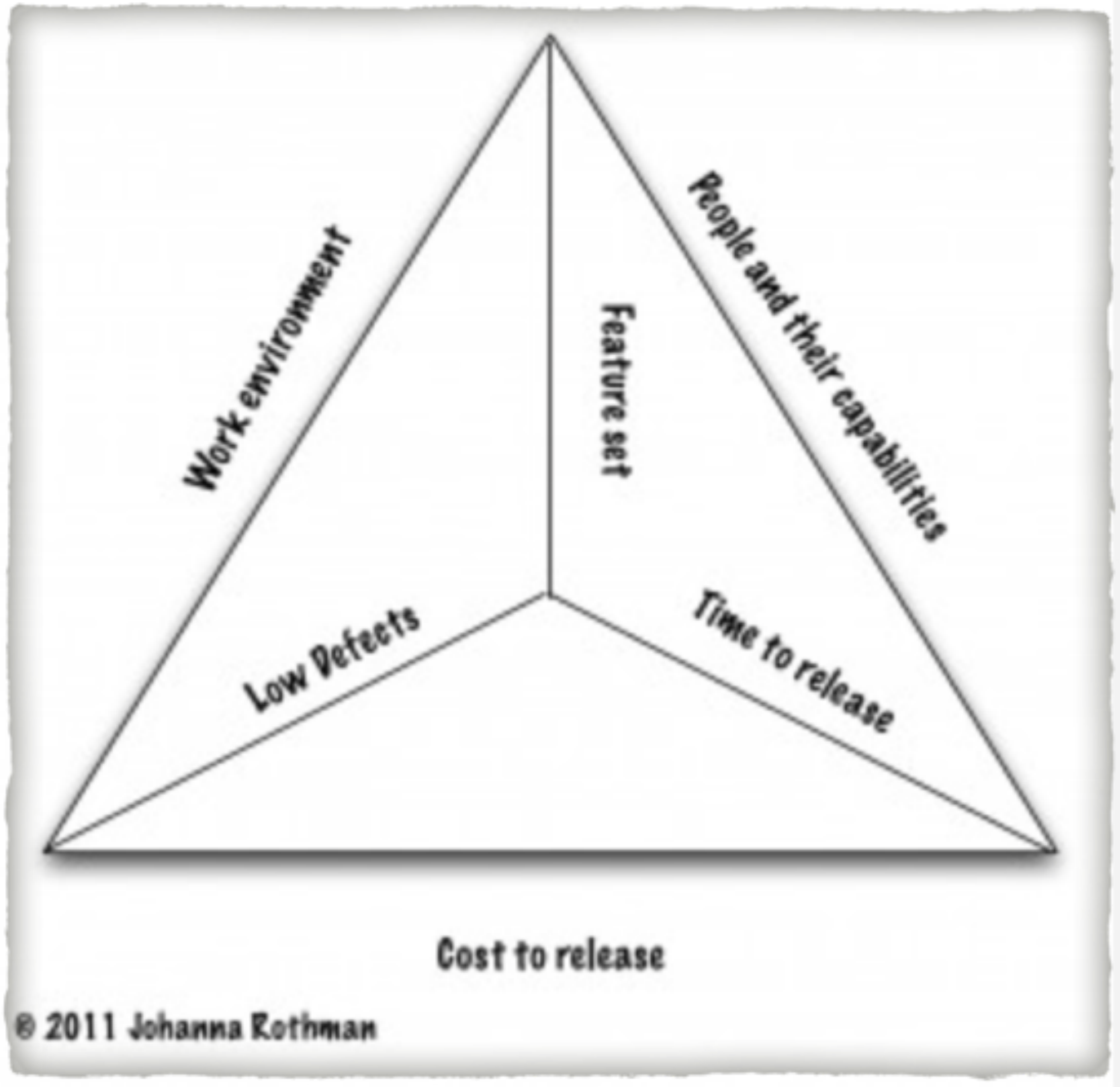
Sizing

- Scope == Sum of story points
    - Scope Buffer of 15-20%
        - Allow for this much churn during development of features.
    - Schedule Buffer - 20%
        - Counter delays in the critical path
    - Other cross-cutting concerns
        - eg. Accessibility - Add another X% based on experience
    - Take account of any tech debt that needs to be addressed.

Iron Triangle - *Relevant?*



New thinking: Project Pyramid [1]



Project Pyramid

- Customers want what is internal to the pyramid.
- Corporate constraints are shown on the outside of the pyramid.

Degrees of freedom?

- Ask business what is priority #1?
  - Features - do all the features?
  - Date? - finish by certain date
  - Cost? - complete within certain costs

Duration

- $\text{Duration} = \dfrac{\text{Effort}}{\text{Velocity}}$
- Use historical velocity data if the team and work are similar.
- Else predict an initial velocity then adjust as iterations proceed.
  - Typically initial velocity is a range

Predicting Initial Velocity

- If the cards vary in size then estimate over 3 iterations.
    - Estimate each card in terms of time.
    - Take an average.
- If the cards are the same size then estimate sample of cards.

Refine over time

- Initial estimates are a range
- Re-plan and adjust using data from iterations
  - If you have a fixed date to release then manage your scope.
- Refine your confidence level over time.
  - Aim for 90% confidence.

What if date does not fit?

- Consider multiple teams - Break up into streams
    - Some SP normalising may be needed
- Do *not* increase the ideal team size
    - Else you end up with too many lines of communication.

Ideal team size

- *7* ± *2* people
  - 3-4 developers - includes Tech Lead
  - 2 Testers (+1 automation)
    - Ratio of testers/developers == 0.5
  - PO, IM, BA, CX, UX
  - Env;Data; Solution Architect ; Backend; Security; Risk; Legal
- Try not to change the team size. Fire up another team if the dates do not fit.

What invalidates using historical velocity ?

- Team members have changed.
- Technology that the team is using has changed.
- Team is moving to a different architecture approach.
- Dependencies are different - one reliant on new services.
- ...

References

- [1] http://www.jrothman.com/mpd/project-management/2011/11/estimating-the-unknown-dates-or-budgets-part-1/, Johanna Rothmans, 2011
- [2] https://www.mountaingoatsoftware.com/blog/its-effort-not-complexity, Mike Cohn, 2010