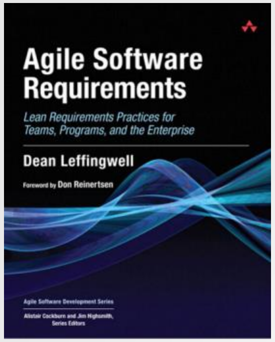


Agile Software Requirements

Lean Requirements Practices for Teams, Programs, and the Enterprise

Dean Leffingwell

Foreword by Don Reinertsen

Agile Software Development Series

Alistair Cockburn and Jim Highsmith, Series Editors

Agile Software Requirements

Software Requirements Engineering – 40688

Computer Engineering department

Sharif university of technology

Fall 402

Chapter 8:

Agile Estimating and Velocity

More Reliable Estimates

We should do two main tasks:

1) **Know where you are now.**
    1) The agile project knows exactly where it is.
    2) It is based on a subjective evaluation of working code
2) **More accurately predict where you will be next.**
    1) They will deliver next based on the *velocity*.

The Business Value of Estimating

1) Determining cost
2) Establishing prioritization
3) Scheduling and commitment.
4) Estimating is the key to unlocking the ability to commit

Estimating Scope with Story Points

Art of relative estimating with story points.

A **story point** is an **integer number** that represents an aggregation of a number of aspects.

1) **Knowledge**: Do we understand what the story does?
2) **Complexity**: How hard is it to implement?
3) **Volume**: How much of it is there? How long is it likely to take?
4) **Uncertainty**: What isn't known, and how might that affect our estimate?

✓ A two-point story should expect to take twice as long as a one-point story.

Relative Estimating

Bigness

1) What does the instructor mean by bigness?
   - Height
   - weight
   - mass
   - muscle
   - bite
   - attitude

2) What the heck kind of poodle is it?
   - Standard poodle?
   - Toy poodle?

   “Hey, it makes a big difference!”

3) What scale should we use?

Estimating Real Work with Planning Poker (1)

- **Participants** include **all agile team members**
- The **product owner** participates but **does not estimate**.
- **Each estimator** is given **a deck of cards** with 0, 1, 2, 3, 5, 8, 13, 20, 40, and 100 as their "**value**."

Estimating Real Work with Planning Poker (2)

- Some amount of **preliminary design discussion** is appropriate. However, **spending too much time** on design discussions is often **wasted effort**.
- **The teams** will be given **a short time box** (maybe 30 minutes) and are instructed **to estimate all the items** on the list within the time box.

Estimating Real Work with Planning Poker (2)

For each story:

1) Product owner reads the description.
2) Questions are asked and answered.
3) Each estimator privately selects a card representing his or her estimate.
4) All cards are simultaneously turned over so that all participants can see each estimate.
5) High and low estimators explain their estimates.
6) After discussion, each estimator reestimates, and the cards are turned over for a second time.
7) The estimates will likely converge. If not, the process for that story is repeated until it does.

- Repeat until all stories are estimated.

Some notes

1) The estimate comes from the team as a whole including developer and testers.

2) The range of numbers is cleverly designed.

3) The expanded range (20, 40, 100) at the end of the series.

4) Zero gives the teams a way to ignore small stories.

5) The cards are turned over all at once.

6) It happens pretty fast.

7) Spending too much time does not generally increase the accuracy of the estimates.

Online Poker Planning

- There are online tools that support planning poker.
- *www.planningpoker.com*
- The team has a separate teleconference or chat session.
- After each vote converges, the moderator presents the next story.

Tabletop Relative Estimation (1)

- Like planning poker, this technique involves the entire team.
- Requires face-to-face communication.
- Proponents of this technique note that it can be faster than planning poker.
- Visualization of the entire iteration enhances the team's understanding.

Tabletop Relative Estimation (2)

1) Team discusses each story in the backlog and places the story on the table.

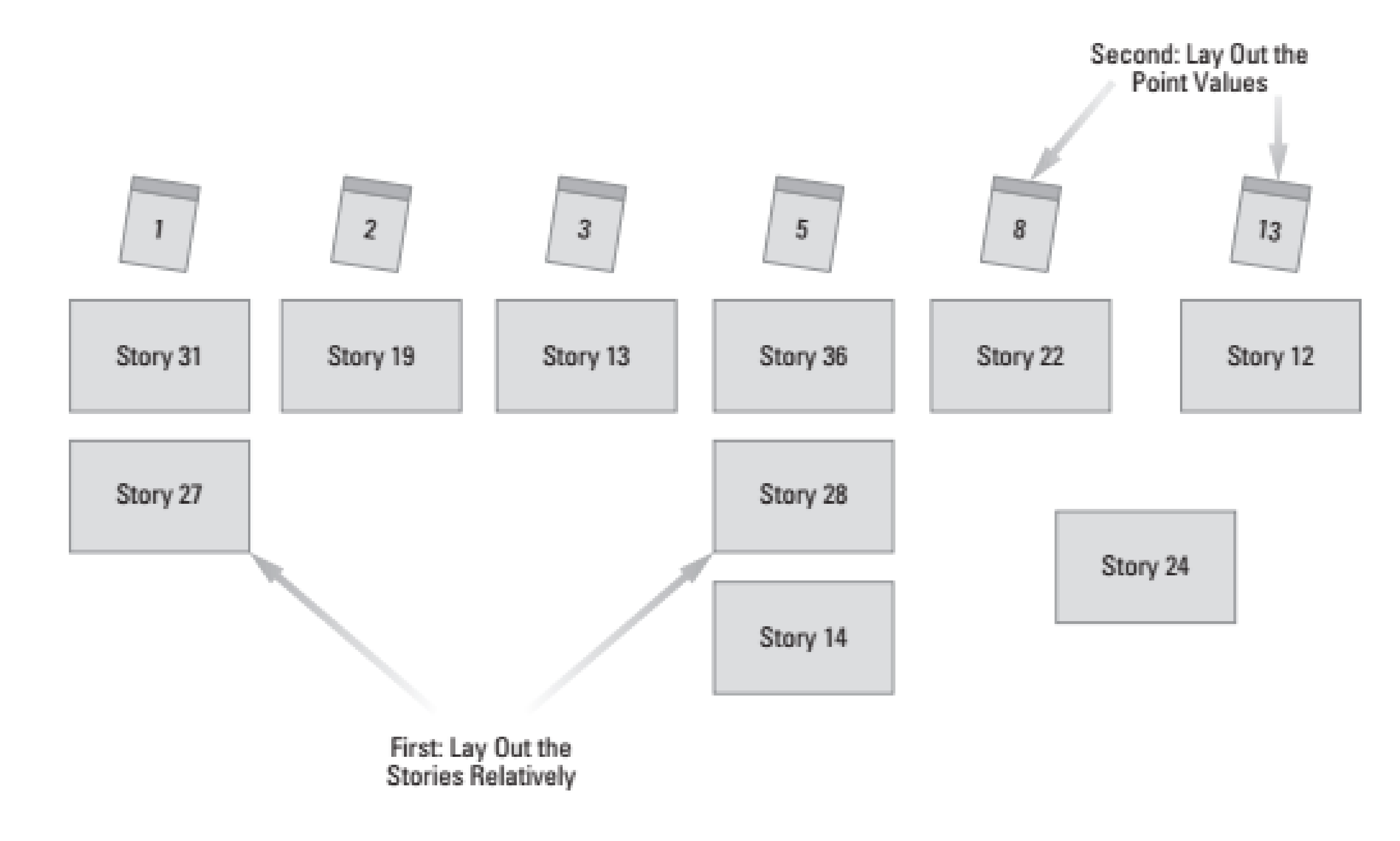
2) Small stories to the left.

3) Bigger stories to the right.

4) Stories of about the same size are stacked in columns.

5) To create the actual estimates, points can be assigned to columns.

Assigning point values to columns of stories



Velocity

- A team's velocity is simply **how many points that team can complete in a standard iteration**.
- Given a team's known historical velocity in a given domain.
- They can now predict how long it will take them.
- Velocity can be a fairly reliable predictor of short-term future events.
- It is not a tool for managing teams.

Caveats on the relative estimating model (1)

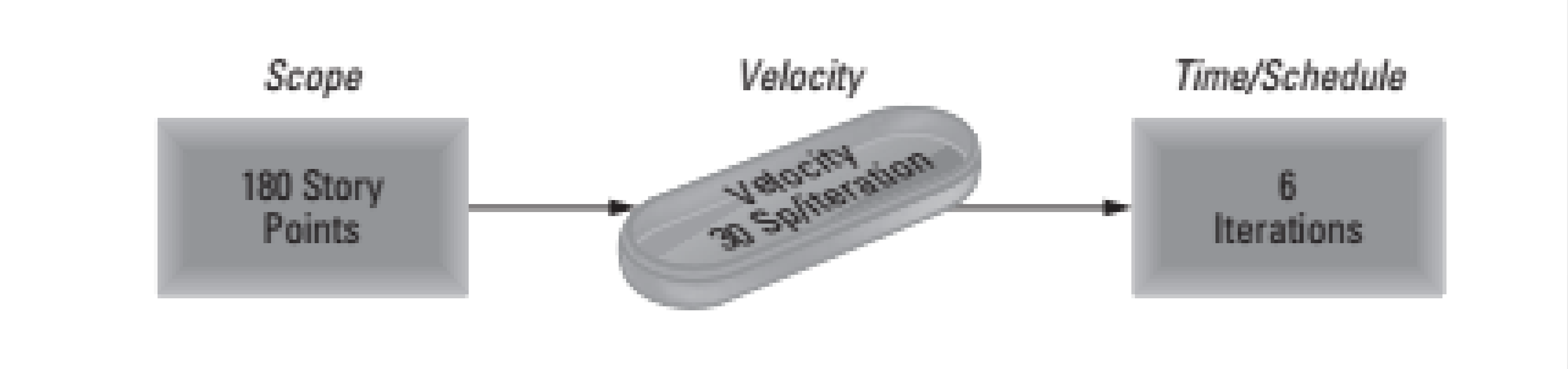
- It is based on historical data and is predictive only to the extent that the future (new stories) looks like the past (stories already completed).
- It is valid only to the extent that the team continues to have the same individuals.
- A team's velocity cannot be compared to any other team.

Caveats on the relative estimating model (2)

- If management attempts to use velocity as a measure of team performance:
  1) Continuously improve the team's true productivity and agility.
  2) Cut back on quality, building technical debt for a future period.
  3) Simply increase the size of the estimates

Estimating schedule

# days to do the work =
# days per iteration * (backlog size estimate / velocity)



Estimating Cost

- Estimating cost to work down a backlog is also fairly readily calculable.
- Take the average burdened cost for a team and divide it by their velocity.
- That provides the cost per story point for that team.
- Multiply the cost per story point for that team by the total estimate for the backlog.

Ideal Developer days

- Story is estimated based on the number of total person days, including development and test, that the team thinks they will need to accomplish the story.
- IDDs are conceptually simpler than story points.

Story Point Concerns

- It isn’t so easy to understand by the team.
- It’s hard to get started.
- Getting to schedule and cost estimates is very indirect.
- Teams occasionally struggle to adjust their velocity based on the availability of team members.
- Team velocities are not normalized.

IDD Concerns

- They don’t have the law of large relative numbers to average.
- It’s far more personal and can be politically loaded.
- It’s the way we used to do it, and heaven knows, that didn’t work very well.

A Hybrid Model

- Teams can proceed in large part with the relative estimating model.
- We add two simple rules:
  - Each team is guided to estimate the smallest story, one that can be done by one person in about a day, as a 1.
  - Each team is also guided to initially estimate that they have eight IDDS per team member per two-week iteration (or adjust accordingly).
  - This leaves about 20% for planning, demoing, company functions, training, and other overhead.

Hybrid Model Advantages

- The teams can still use planning poker.
- The estimate is still a consensus.
- They can start immediately.
- The relative methods still avoid any tendency to overinvest in estimating.
- The translation to cost is obvious.