An Introduction to Agile Estimating and Planning

Mike Cohn
12 March 2008

Mike Cohn - background

- Agile coach and trainer
- Founding member and director of Agile Alliance and Scrum Alliance
- Founder of Mountain Goat Software
- Ran my first Scrum project back in 1995
- Typical programmer to manager; etc. progression
Agenda

- The right units for estimating
- How to estimate
- Release planning
- Planning with multiple teams

How long will it take…

- …to read the latest Harry Potter book?
- …to drive to Paris?
Estimate size; derive duration

Size  Calculation  Duration

300 kilograms  Velocity = 20  300/20 = 15 sprints

Measures of size

**Sequential**
- Lines of code
- Function points

**Agile**
- Story points
- Ideal days
Ideal days

- How long something would take if
  - it’s all you worked on
  - you had no interruptions
  - and everything you need is available
- The ideal time of a basketball game is 40 minutes
  - Four 10-minute quarters
  - The elapsed time is much longer (2+ hours)

Story points

- The “bigness” of a task
  - Influenced by
    - How hard it is
    - How much of it there is
  - Relative values are what is important:
    - A login screen is a 2.
    - A search feature is an 8.
  - Points are unit-less

As a user, I want to be able to have some but not all items in my cart gift wrapped.
Zoo points

What value in “zoo points” would you put on these zoo animals?

Lion
Kangaroo
Rhinoceros
Bear
Giraffe
Gorilla
Hippopotamus
Tiger

Comparing the approaches

• Story points help drive cross-functional behavior
• Story point estimates do not decay
• Story points are a pure measure of size
• Estimating in story points is typically faster
• My ideal days cannot be added to your ideal days
• Ideal days are easier to explain outside the team
• Ideal days are easier to estimate at first
• Ideal days can force companies to confront time wasting activities
The problem with mixing units

As a frequent flyer, I want…
- Code the… 12
- Design the… 10
- Automate… 5
- Test the… 8

<table>
<thead>
<tr>
<th>Task</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>As a frequent flyer, I want…</td>
<td>30</td>
</tr>
<tr>
<td>As a frequent flyer, I want…</td>
<td>20</td>
</tr>
<tr>
<td>As a frequent flyer, I want…</td>
<td>60</td>
</tr>
<tr>
<td>As a frequent flyer, I want…</td>
<td>40</td>
</tr>
<tr>
<td>As a frequent flyer, I want…</td>
<td>20</td>
</tr>
</tbody>
</table>

Agenda

- The right units for estimating
- How to estimate
- Release planning
- Planning with multiple teams
Estimate by analogy

- Comparing a user story to others
  - “This story is like that story, so its estimate is what that story’s estimate was.”
- Don’t use a single gold standard
  - Triangulate instead
  - Compare the story being estimated to multiple other stories

Use the right units

- Can you distinguish a 1-point story from a 2?
  - How about a 17 from an 18?
- Use a set of numbers that make sense; I like:
  - 1, 2, 3, 5, 8, 13
  - Stay mostly in a 1-10 range
- Nature agrees:
  - Musical tones and volume are distinguishable on a logarithmic scale

Use 0 and \( \frac{1}{2} \) if you like
Planning poker

- An iterative approach to estimating
- Steps
  - Each estimator is given a deck of cards, each card has a valid estimate written on it
  - Customer/Product owner reads a story and it's discussed briefly
  - Each estimator selects a card that's his or her estimate
  - Cards are turned over so all can see them
  - Discuss differences (especially outliers)
  - Re-estimate until estimates converge

Planning poker—an example

<table>
<thead>
<tr>
<th>Estimator</th>
<th>Round 1</th>
<th>Round 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erik</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Martine</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Inga</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Tor</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>
Estimate these

<table>
<thead>
<tr>
<th>Product backlog item</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read a high-level, 10-page overview of agile software development in a celebrity magazine.</td>
<td></td>
</tr>
<tr>
<td>Read a densely written 5-page research paper about agile software development in an academic journal.</td>
<td></td>
</tr>
<tr>
<td>Write the product backlog for a simple eCommerce site that sells only clocks.</td>
<td></td>
</tr>
<tr>
<td>Recruit, interview, and hire a new member for your team.</td>
<td></td>
</tr>
<tr>
<td>Create a 60-minute presentation about agile estimating and planning for your coworkers.</td>
<td></td>
</tr>
<tr>
<td>Wash and wax your boss’ Porsche.</td>
<td></td>
</tr>
<tr>
<td>Read a 150-page book on agile software development.</td>
<td></td>
</tr>
<tr>
<td>Write an 8-page description of agile development for your boss.</td>
<td></td>
</tr>
</tbody>
</table>

www.planningpoker.com

Free, or I wouldn't mention it.
Agenda

- The right units for estimating
- How to estimate
- Release planning
- Planning with multiple teams

Release planning

Release Planning Meeting

Release Plan

Sprint 1 | Sprint 2 | Sprint 3 | Sprints 4–7
An example with velocity = 14

Sprint 1
Story A 5
Story B 8
Story E 1

Sprint 1
Story C 3
Story D 5
Story F 3
Story G 3
Story H 13
Story I 5
Story J 8

Projections based on velocity

Mean (Best 3) = 37
Mean (Last 8) = 33
Mean (Worst 3) = 28
Extrapolate from velocity

Assume 5 sprints left

At our slowest velocity, we'll end here (5 x 28)
At our average velocity, we'll end here (5 x 33)
At our average velocity, we'll end here (5 x 37)

Fixed-date planning

How much can I get by <date>?

- Determine how many sprints you have
- Estimate velocity as a range
- Multiply low velocity x number of sprints
  - Count off that many points; These are “Will Have” items
- Multiply high velocity x number of sprints
  - Count off that many more points; these are “might haves”
### Fixed-date planning example

<table>
<thead>
<tr>
<th>Desired release date</th>
<th>30 June</th>
</tr>
</thead>
<tbody>
<tr>
<td>Today’s date</td>
<td>1 January</td>
</tr>
<tr>
<td>Number of sprints</td>
<td>6 (monthly)</td>
</tr>
<tr>
<td>Low velocity</td>
<td>15</td>
</tr>
<tr>
<td>High velocity</td>
<td>20</td>
</tr>
</tbody>
</table>

- **Will have**: $6 \times 15 = 90$
- **Might have**: $6 \times 20 = 120$
- **Won’t have**

### Agenda

- The right units for estimating
- How to estimate
- Release planning
- Planning with multiple teams
Three issues

1. Estimating in a common unit
2. Sprint planning
3. Dependencies

Establish a common baseline

- All teams should agree on story points or ideal days
- Establish a common baseline
  - Select a dozen or so user stories that were done recently or are on the product backlog
  - Estimate them en masse with Planning Poker
Be careful with cross-team comparisons

- When did this firm start comparing velocity?
- When did the yellow team figure out they were being compared?

Two approaches to sprint planning

1. **Stagger by a day**

- Sprints end by ± a day
- Helps a key resource (e.g., a product owner or architect) fully participate in many planning meetings
The Big Room

- All sprints end on same day
- All planning is on same day and in one room
- Key resources shift between teams on demand

Dependencies

- Critical dependencies between teams
  - Must be done in this order and likely to influence overall ship date
  - Fewer of these than you may think
- Emergent dependencies
  - “OK, we’re going to start on such-and-such soon. As you know we need this-and-that first.”
Buffer critical dependencies

Sprint 1 | 20 points
---|---
Sprint 2 | 10 points
Sprint 3 | 20 points

Sprint 1 | 17 points
---|---
Sprint 2 | 17 points
Sprint 3 | 17 points

Rolling lookahead planning

<table>
<thead>
<tr>
<th>Task</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code the …</td>
<td>8</td>
</tr>
<tr>
<td>Test the …</td>
<td>16</td>
</tr>
<tr>
<td>Integrate with …</td>
<td>5</td>
</tr>
<tr>
<td>Code the …</td>
<td>8</td>
</tr>
<tr>
<td>Design the …</td>
<td>4</td>
</tr>
</tbody>
</table>

© 2003–2007 Mountain Goat Software®
After Sprint 1

Sprint 1
Code... 8
Test... 4
Design 4
Code... 5

Sprint 2
Sprint 3
Sprints 4–7

After Sprint 2

Sprint 2
Code... 3
Test... 7
Test... 6
Code... 8

Sprint 3
Sprint 4
Sprints 5–7

• While planning Sprint 2, a team rolls Sprint 4 into view.
• They discover a dependency on another team.
• The other team work on that item during Sprint 3.

Mike Cohn contact info

Agile Estimating and Planning

Mike Cohn

User Stories Applied
For Agile Software Development

mike@mountaingoatsoftware.com
www.mountaingoatsoftware.com
(720) 890-6110

© 2003–2007 Mountain Goat Software®