An Introduction to Agile Estimating and Planning

Mike Cohn
February 28, 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

© Mountain Goat Software, LLC
Agenda

☐ The right units for estimating
☐ How to estimate
☐ Release planning
  ☐ Fixed-date projects
  ☐ Planning with multiple teams

How long will it take...

• ...to read the latest Harry Potter book?
• ...to drive to Austin?
Estimate size; derive duration

Size → Calculation → Duration

300 kilograms → Velocity = 20 → 300/20 = 15 sprints

Measures of size

- Traditional and agile measure size differently

Traditional measures of size
- Lines of Code
- Function Points

Agile measures of size
- Story points
- Ideal days
**Ideal time**

- 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 football game is 60 minutes
  - Four 15-minute quarters
- The elapsed time is much longer (3+ 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
Dog points

What value in “dog points” would you put on these breeds?

Labrador retriever
Dachshund
Great Dane
German Shepherd
Poodle
St. Bernard

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

Product Backlog

<table>
<thead>
<tr>
<th>As a frequent flyer, I want to...</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>As a frequent flyer, I want to...</td>
<td>50</td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td>50</td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td>20</td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td>20</td>
</tr>
</tbody>
</table>

Sprint Backlog

<table>
<thead>
<tr>
<th>Code the UI</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write test fixture</td>
<td>6</td>
</tr>
<tr>
<td>Code middle tier</td>
<td>12</td>
</tr>
<tr>
<td>Write tests</td>
<td>5</td>
</tr>
<tr>
<td>Automate tests</td>
<td>4</td>
</tr>
</tbody>
</table>

“Yesterday I started on the UI; I should finish before the end of today.”

Agenda

- The right units for estimating
- How to estimate
- Release Planning
  - Fixed-date projects
  - 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

Triangulation

- Confirm estimates by comparing the story to multiple other stories.
- Group like-sized stories on table or whiteboard
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

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>Susan</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Vadim</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Ann</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Chris</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

### Agenda

- ✔️ The right units for estimating
- ✔️ How to estimate
- ☐ Release planning
  - ☐ Fixed-date projects
  - ☐ 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 C: 3
Story D: 5
Story E: 1

Sprint 2

Story F: 5
Story G: 1
Story H: 13
Story I: 5
Story J: 8
Projections from velocity

- Mean (Best 3) = 37
- Mean (Last 8) = 33
- Mean (Worst 3) = 28

Extrapolate from velocity

Assume: There are five sprints left.

- At our slowest velocity we’ll finish here (5×28)
- At our long-term average we’ll finish here (5×33)
- At our best velocity we’ll finish here (5×37)

Consider drawing an arrow at most recent velocity as well.
Fixed-date planning

How much can I get by <date>?

1. Determine how many sprints you have
2. Estimate velocity as a range
3. Multiply low velocity × number of sprints
   - Count off that many points
   - These are “Will Have” items
4. Multiply high velocity × number of sprints
   - Count off that many more points
   - These are “Might Have” items

Fixed-date planning: an 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 × 15
Might have: 6 × 20
Won’t have
Planning with multiple teams

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>Tasks</th>
<th>Est</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 the...</td>
<td>8</td>
</tr>
<tr>
<td>Code the ...</td>
<td>12</td>
</tr>
<tr>
<td>Design the ....</td>
<td>8</td>
</tr>
</tbody>
</table>
Suppose that while planning sprint 2, an item that rolls into visibility for sprint 4 is dependent on work by another team.

Another team can work to fulfill the dependency during sprint 3.

Mike Cohn contact info

mike@mountaingoatsoftware.com
www.mountaingoatsoftware.com
(720) 890-6110 (office)
(303) 810-2190 (mobile)