Mike Cohn - background

Consultant, author, and speaker

Founding member and director of Agile Alliance
Agile Coach International
Scrum Master Trainer

Founder of Mountain Goat Software
-Process and project management training and consulting
Today’s agenda

- Overview
- Estimating in story points
- Estimating in ideal time
- Techniques for estimating
- Iteration planning
- Release planning
- Estimating velocity

What’s a good plan?

- A good plan is one that supports reliable decision-making
- Will go from
  - We’ll be done in the fourth quarter
  - We’ll be done in November
  - We’ll be done November 7th

Copyright Mountain Goat Software, LLC
What makes planning agile?

- Is more focused on planning than the plan
- Encourages change
- Results in plans that are easily changed
- Is spread throughout the project

An agile approach to planning
Estimates on the product backlog

<table>
<thead>
<tr>
<th>Iteration 1</th>
<th>Product Backlog</th>
<th>Iteration Backlog</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>Code the UI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>Write test fixture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>Code middle tier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>Write tests</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>Automate tests</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Iteration 2</th>
<th>Product Backlog</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We’re talking about these right now
How long will it take...

- ...to read the latest Harry Potter book?
- ...to drive to Seattle?

Estimate size; derive duration

Size: 300 pounds
Calculation: Velocity = 20
Duration: 300 / 20 = 15 iterations
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

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.

5
Dog points

Assign “dog points” to the following breeds:

- Labrador retriever
- Dachshund
- Great Dane
- Terrier
- German Shepherd
- Poodle
- St. Bernard
- Bulldog

Estimating in Ideal Time
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 basketball game is 40 minutes
  - Four 10-minute quarters
  - The elapsed time is much longer (3+ hours?)

Elapsed time vs. ideal time

Ideally
- Monday has 8 hours
- Each week has 40 hours

But instead
- 3 hours of meetings
- 1 hour of email
- 4 hours left for the project

"How long will this take?"
- Are you answering what is being asked?

So, this developer will only make four hours of progress on Monday.

It will take two calendar days to complete one ideal day of work.
Ideal time vs. elapsed time

- It’s easier to estimate in ideal time
- It’s too hard to estimate directly in elapsed time
- Need to consider all the factors that affect elapsed time at the same time you’re estimating

Specialization

- First, don’t worry about it too much
  - We’re usually better off with fairly rapid, imprecise estimates than spending more time
- Second
  - Just add up the components and report one total estimate of ideal days

\[ = 8 \\
= 8 \\
= 8 \]
The great debate

Story points or ideal days?

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
What I usually do

- I prefer story points
  - ...but they make some teams uncomfortable, so I’ll
- Start with ideal time
  - Gives the team a nice foundation for the initial stories
  - Helps team get started
- Define “1 story point = 1 ideal day” then
  - Gradually convert team to thinking in unit-less story points
  - “This story is like that story.”
  - Stop talking about how long it will take

Three levels of planning...
...three levels of precision

<table>
<thead>
<tr>
<th>Product Backlog</th>
<th>Iteration Backlog</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Iteration 1</strong></td>
<td></td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td>Code the UI 8</td>
</tr>
<tr>
<td></td>
<td>Write test fixture 6</td>
</tr>
<tr>
<td></td>
<td>Code middle tier 12</td>
</tr>
<tr>
<td></td>
<td>Write tests 5</td>
</tr>
<tr>
<td></td>
<td>Automate tests 4</td>
</tr>
<tr>
<td><strong>Iteration 2</strong></td>
<td></td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>As a frequent flyer, I want to...</td>
<td></td>
</tr>
</tbody>
</table>

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

---

Techinques for Estimating
Approaches to estimating

- Analogy
- Disaggregation
- Planning poker

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

Disaggregation

- Breaking a big story into smaller stories or tasks
- Sometimes very useful
- But disaggregating too far causes problems
  - Forgotten tasks
  - Summing lots of small errors can be big number
How much effort?

- A little effort helps a lot
- A lot of effort only helps a little more

Use the right units

- Can you distinguish a 1-point story from a 2?
- Can you distinguish a 17 from an 18?
- Use units that make sense, such as
  - 1, 2, 3, 5, 8
  - 1, 2, 4, 8
- Stay mostly in a 1-10 range

Include 0 and ½ if you want
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>
Remodeling my kitchen

1. Install new hardwood floor
2. Refinish (remove, sand, repaint) the cabinets
3. Replace my tile countertop with granite
4. Repaint entire kitchen
5. Lay shelf paper
6. Install recessed lighting
7. Replace electric stove with gas stove
8. Install built-in refrigerator
9. Install a new oven
10. Plumb the island and add sink

Why planning poker works

- Emphasizes relative estimating
- Focuses most estimates within an approximate one order of magnitude
- Everyone’s opinion is heard
- Estimators are required to justify estimates
- It’s quick
- It’s fun
**Iteration Planning**

**Product Backlog**

<table>
<thead>
<tr>
<th>Iteration 1</th>
<th>As a frequent flyer, I want to...</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>5</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>As a frequent flyer, I want to...</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>As a frequent flyer, I want to...</td>
<td>2</td>
</tr>
</tbody>
</table>

**Iteration Backlog**

<table>
<thead>
<tr>
<th></th>
<th>Iteration Backlog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code the UI</td>
<td>8</td>
</tr>
<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."

**Creating this is iteration planning**
Two approaches

- Velocity-driven iteration planning
  - “We finished 15 story points last time, let’s plan on 15 story points this time.”
- Commitment-driven iteration planning

Commitment-driven iteration planning

- Discuss the highest priority item on the product backlog
- Decompose it into tasks
- Estimate each task
  - Whole team estimates each task
- Ask ourselves, “Can we commit to this?”
  - If yes, see if we can add another backlog item
  - If not, remove this item but see if we can add another smaller one
Estimate availability

<table>
<thead>
<tr>
<th>Person</th>
<th>Hours per Day</th>
<th>Hours per Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sergey</td>
<td>4-6</td>
<td>40-60</td>
</tr>
<tr>
<td>Yuri</td>
<td>5-7</td>
<td>50-70</td>
</tr>
<tr>
<td>Carina</td>
<td>2-3</td>
<td>20-30</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>110-160</strong></td>
</tr>
</tbody>
</table>

It looks something like this

- Code the abc class (8 hours)
- Code the user interface (4)
- Write test fixtures (4)
- Code the xyz class (6)
- Update performance tests (4)
- Prototype the UI (8 hours)
- Demo UI to 3 outside users (3)
- Code new UI (12)
- Update documentation (3)
Predicting velocity

- Three ways to come up with a velocity:

  - Use historicals
  - Run a few iterations
  - Forecast it

- Express velocity as a range that matches your uncertainty in it
Forecasting velocity

- Just like commitment-driven iteration planning
- Estimate available hours for the iteration
- Repeat until full:
  - Pick a story, break into tasks, estimate each task

Ideally, “plan” more than one iteration

An example

| Code the UI | 8 |
| Write tests | 6 |
| Code middle tier | 12 |
| Write tests | 5 |

| Code the ... | 12 |
| Do the... | 8 |
| Document the... | 8 |
| Test the... | 8 |
| Do the... | 4 |
| Document the... | 8 |

As a frequent flyer, I want to...
As a user, I want to...
As a vacation planner, I want to...
As a frequent flyer, I want to...

At 110-160 available hours per iteration, what is the team’s velocity?
An example with velocity = 14

Updating the release plan

- Revisit the release plan at the end of every sprint
- Update it based on:
  - Current understanding of velocity
  - Current prioritization of the product backlog
- This should be a very short and sweet process
A simple updating example

<table>
<thead>
<tr>
<th>Story</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>3</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>G</td>
<td>3</td>
</tr>
<tr>
<td>I</td>
<td>3</td>
</tr>
<tr>
<td>H</td>
<td>5</td>
</tr>
<tr>
<td>J</td>
<td>2</td>
</tr>
<tr>
<td>K</td>
<td>5</td>
</tr>
<tr>
<td>L</td>
<td>3</td>
</tr>
</tbody>
</table>

✓ Story A 5
✓ Story B 3
✓ Story C 5
✓ Story F 3
✓ Story D 5
✓ Story E 5
✓ Story G 3
✓ Story I 3
✓ Story H 5
✓ Story J 2
✓ Story K 5
✓ Story L 3

Look at velocity in a few ways

Velocity

- Last Observation = 36
- Mean (Last 8) = 33
- Mean (Worst 3) = 28
Extrapolate from velocity

At our slowest velocity we’ll finish here
At current velocity we’ll finish here
At our long-term average we’ll finish here

Copyright Mountain Goat Software, LLC

Updating the release plan

Here are the results of the last 8 iterations. There are 6 iterations left. Using this data, update the release plan on the following slide by drawing three arrows into it.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>14</td>
</tr>
</tbody>
</table>

Mean of Worst 3 = 14
Most Recent = 14
Long-term Average = 14

Copyright Mountain Goat Software, LLC
Update this release plan

<table>
<thead>
<tr>
<th>Running Total</th>
<th>Estimate</th>
<th>Story</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>23</td>
<td>13</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>31</td>
<td>8</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>51</td>
<td>20</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>59</td>
<td>8</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>64</td>
<td>5</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>72</td>
<td>8</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>77</td>
<td>5</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>85</td>
<td>8</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>90</td>
<td>5</td>
<td>As a user, I can...</td>
</tr>
<tr>
<td>93</td>
<td>3</td>
<td>As a user, I can...</td>
</tr>
</tbody>
</table>

Upcoming public classes

<table>
<thead>
<tr>
<th>Date</th>
<th>What</th>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>August 1-2</td>
<td>Certified ScrumMaster</td>
<td>Denver</td>
</tr>
<tr>
<td>August 3</td>
<td>Agile Estimating &amp; Planning</td>
<td>Denver</td>
</tr>
<tr>
<td>November 7-8</td>
<td>Certified ScrumMaster</td>
<td>Santa Clara</td>
</tr>
<tr>
<td>November 9</td>
<td>Agile Estimating &amp; Planning</td>
<td>Santa Clara</td>
</tr>
<tr>
<td>November 29-30</td>
<td>Certified Product Owner (with Ken Schwaber)</td>
<td>Boulder</td>
</tr>
</tbody>
</table>

Register at www.mountaingoatsoftware.com
Mike Cohn contact info

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