

# Agile Planning

Mike Cohn

19 June 2009

1

## What's a good plan?

- A good plan is one that supports reliable decision-making
- Will go from
  - We'll be done in the third quarter
  - We'll be done in August
  - We'll be done August 18th

"It's better to be roughly right than precisely wrong."

~John Maynard Keynes



# 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



© 2003–2009 Mountain Goat Software®

3

# Starting assumptions

- We have a product backlog (prioritized feature list)
- Items on the product backlog (user stories is my preference) have been estimated in story points or ideal days
- We have a team with a known velocity
  - We'll relax this assumption shortly

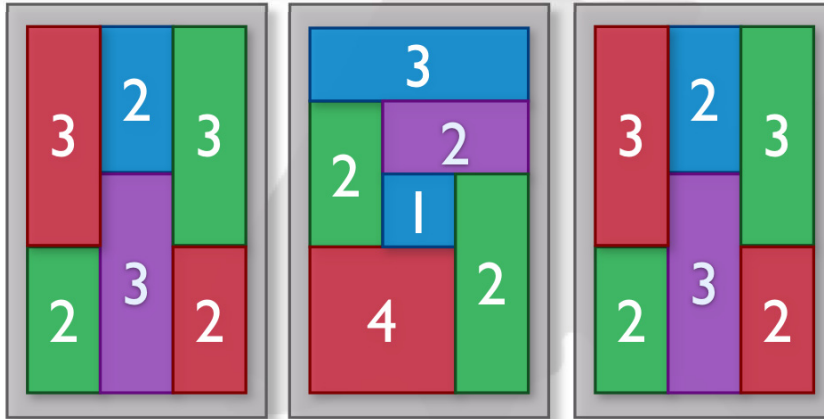


© 2003–2009 Mountain Goat Software®

4

## Velocity

The amount of work planned or completed in an iteration. Measured in the estimating unit used on product backlog items (usually *story points* or *ideal days*).



Three Sprints (Iterations)



© 2003–2009 Mountain Goat Software®

## An example with velocity=14

Sprint 1

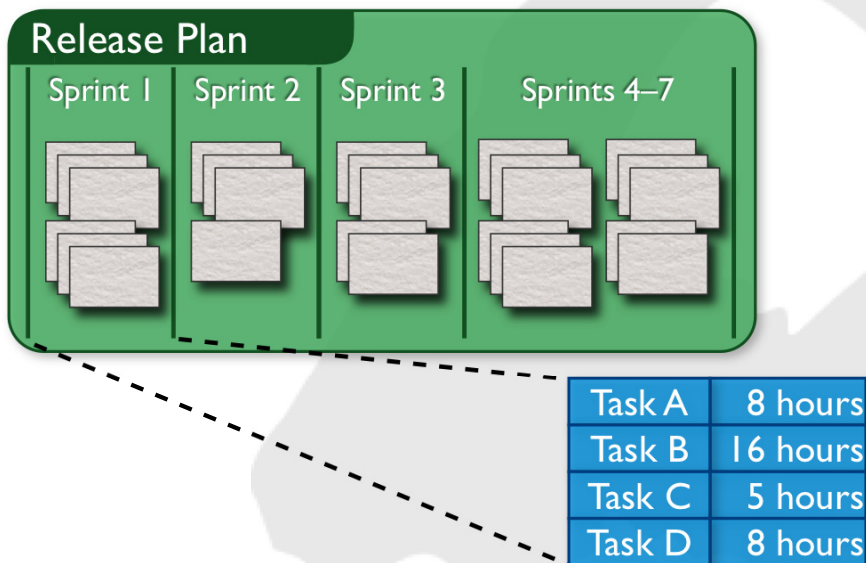
Sprint 2

Story A 5	Story F 5
Story B 8	Story G 1
Story C 3	Story H 13
Story D 5	Story I 5
Story E 1	Story J 8



© Mountain Goat Software, LLC

# Release and sprint planning



## Agenda

- Sprint planning
- Estimating velocity
- Release planning







9

## Two approaches

1

### Velocity-driven sprint planning

- “We finished 15 story points last time, let’s plan on 15 story points this time.”
- Very unreliable in what will be accomplished during an iteration
  - Velocity is mostly useful over the long term



10

2

## Commitment-driven sprint 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



## It looks something like this

As a user, I want ...

2

- Code the abc class (8 hours)
- Code the user interface (4)
- Write test fixtures (4)
- Code the xyz class (6)
- Update performance tests (4)

Team can commit, so they continue...

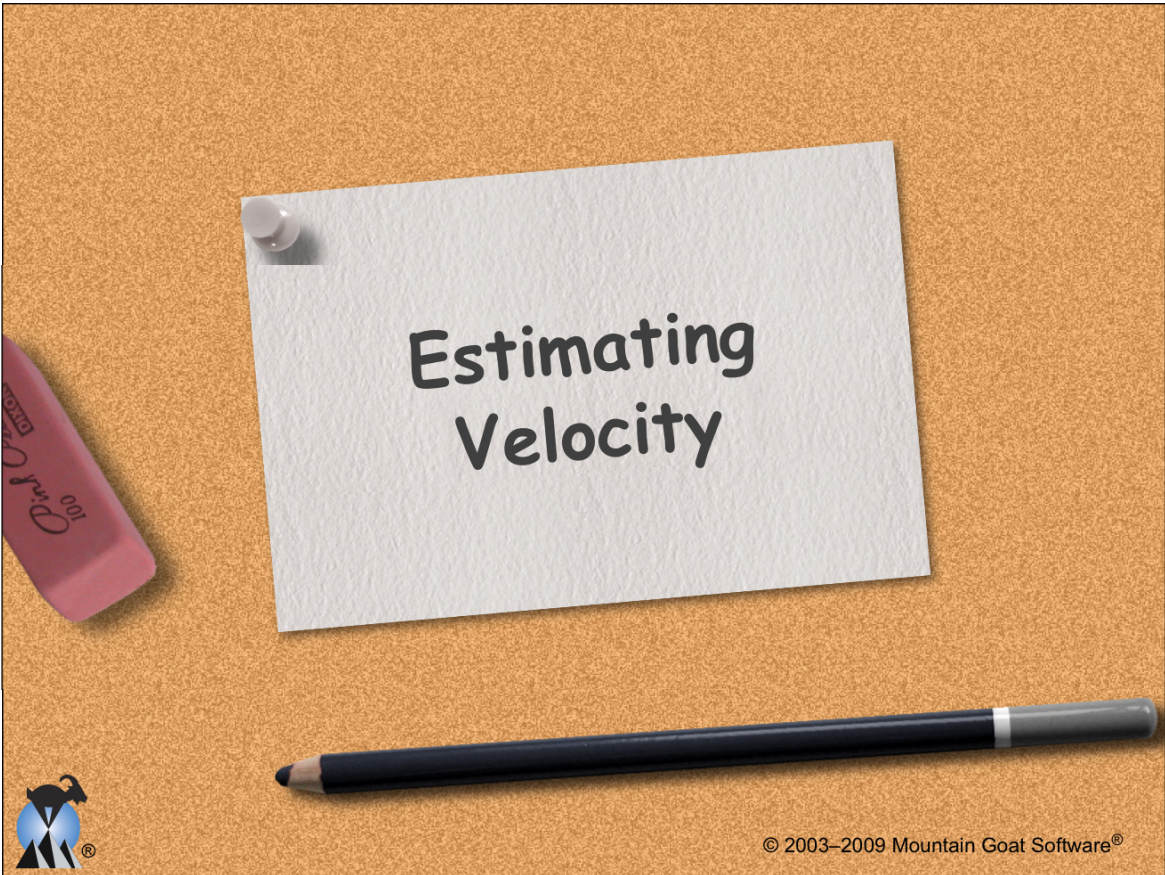
As a user, I want ...

3

- Prototype the UI (8 hours)
- Demo UI to 3 outside users (3)
- Code new UI (12)
- Update documentation (3)







# How to estimate velocity

- 1 Use historical values
- 2 Don't, until you've run a sprint or two
- 3 Forecast it



# Forecasting velocity

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



## An example

- Estimating available hours

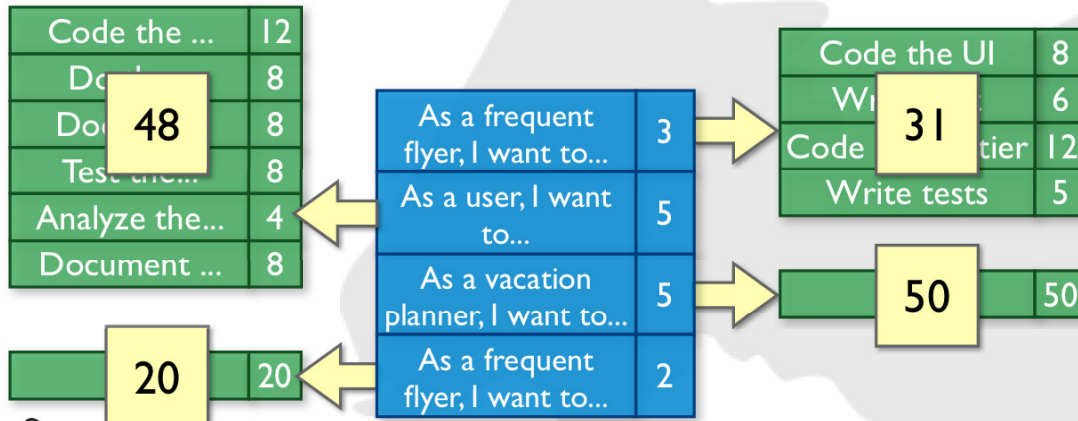
Person	Hours per Day	Hours per Sprint
Sergey	4-6	40-60
Yuri	5-7	50-70
Carina	2-3	20-30
Total		110-160





# An example

At 110-160 available hours per sprint, what is the team's velocity?



© 2003–2009 Mountain Goat Software®

17

# Put a range around it

- You're unlikely to have precisely forecasted the exact velocity the team will average
- So, put a range around your velocity estimate:

Known team, domain, technology	+5% -10%
↕	+10% -25%
Unknown team, domain, technology	+25% -50%

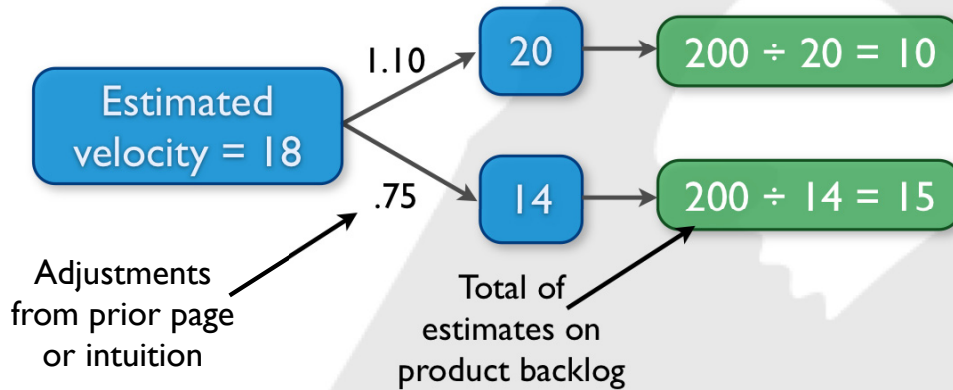


†Numbers based on PMI advice on progressive accuracy of estimates.

© 2003–2009 Mountain Goat Software®

18

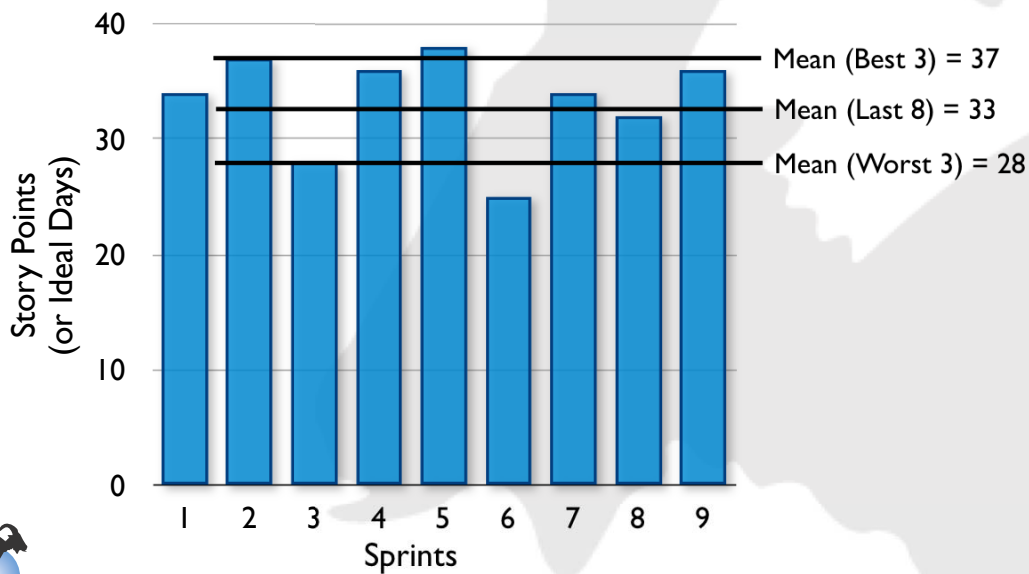
# Expressing velocity as a range



“Right now, before we start this project, our best estimate is that it will take between 10 and 15 sprints.”



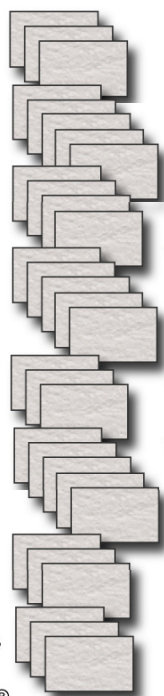
# Create a range from historical data



© 2003–2009 Mountain Goat Software®

21

# Predicting where we'll finish



Assume:  
There are five  
iterations left.

← At our slowest velocity we'll finish here ( $5 \times 28$ )

← At our long-term average we'll finish here ( $5 \times 33$ )

← At our best velocity we'll finish here ( $5 \times 37$ )



© 2003–2009 Mountain Goat Software®

22

# 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”

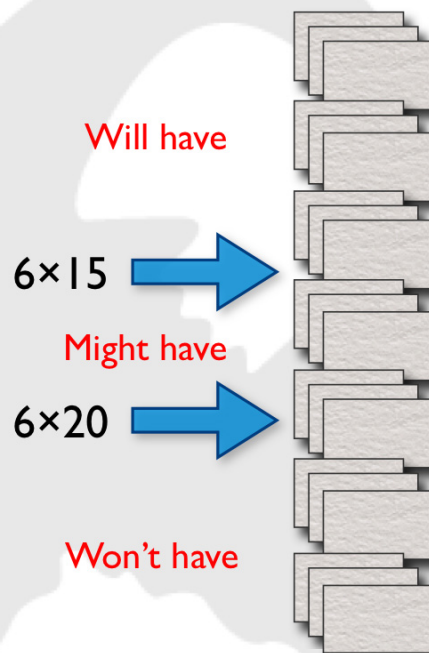


© 2003–2009 Mountain Goat Software®

23

## Fixed-date planning: an example

Desired release date	30 June
Today's Date	1 January
Number of sprints	6 (monthly)
Low velocity	15
High velocity	20

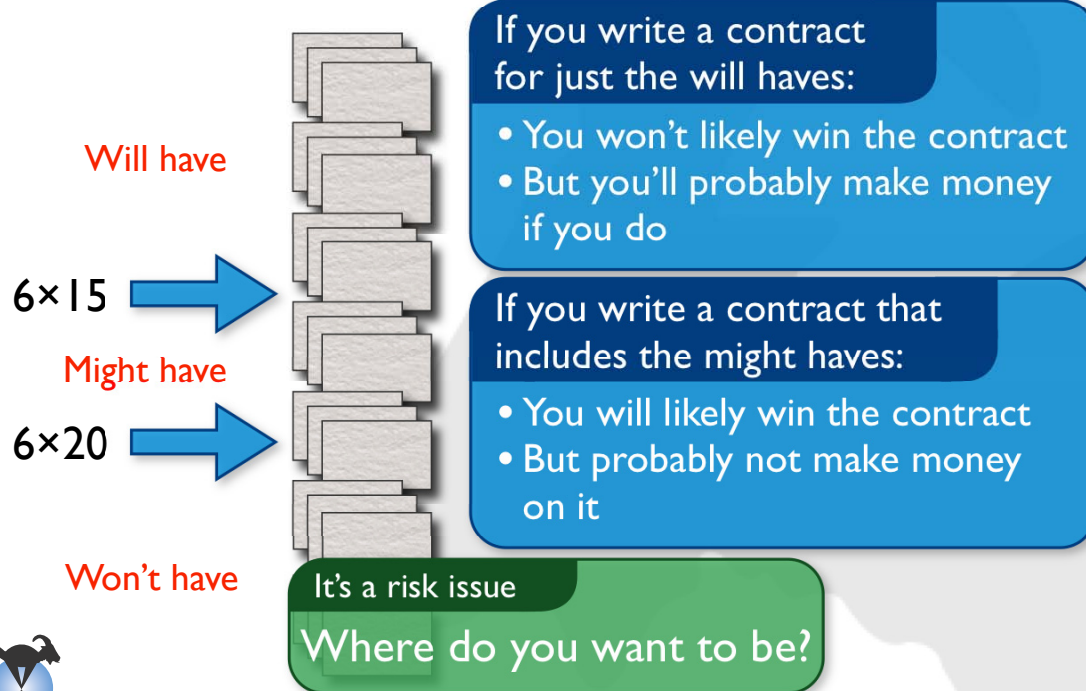


© 2003–2009 Mountain Goat Software®

24



# Fixed-date contracting



# Fixed-scope planning

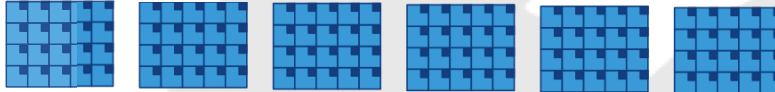
When will all of this be done?

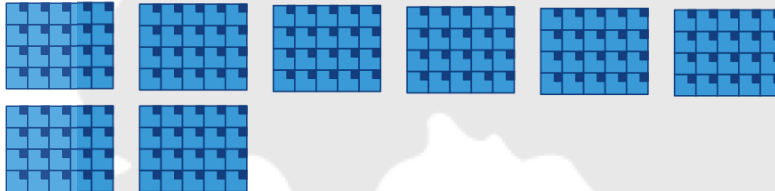
1. Sum all the backlog items the customer *needs*
2. Estimate velocity as a range
3. Divide total story points by high velocity
  - This is the shortest number of sprints it could take
4. Divide total story points by low velocity
  - This is the “most” sprints it could take



# Fixed-scope planning: an example

Total story points desired	120
Low velocity	15
High velocity	20

$120 \div 20 =$  

$120 \div 15 =$  



# Fixed-scope contracting

If you write a contract for the **short** duration:

- You'll likely win the contract
- But may not make any money

If you write a contract for the **long** duration:

- You probably won't win the contract
- But will make money if you

It's a risk issue

Where do you want to be?



# Ranges

- Notice in both cases we had a range
- For a fixed date project, use a scope range:
  - “By that date you’ll have all of these features and some of these.”
- For a fixed-scope project, use a date range:
  - “It will take us between 5 and 8 sprints to deliver all of those features.”



## An example

Your company develops tools for managing agile projects.

You’ve finished version 1.0 (on time, of course).

Now the boss needs a new version for the big trade show that is 4 sprints away.

- Which features can you “guarantee” will be in for the trade show?
- Which features are likely to be in?



# Past velocities

Historical Data	
Iteration number	Velocity
1	20
2	14
3	23
4	18
5	25
6	30
7	12
8	22
9	15
10	23

Your Estimates	
Low	
Average	
High	



# The team's estimates



Product backlog item	Estimate
As the product owner I want to drag items onto a release burndown chart and see the impact to the release date.	20
As a user at a company with lots of cash, I want your product to support touch screens so I can put a large one in our team room.	13
As a user I would like performance to be about twice as fast as now during peak use periods.	20
As a team member, I'd like to be able to do online planning poker estimating right inside the tool.	13
As a third party, I would like an SOA interface so that I can integrate my product with yours.	8
As a team member I want RSS support for all changes to tasks or user stories so that I'm notified.	8
As the product owner, I want a new report that shows differences in the product backlog between different time periods.	3
As a team member I'd like to define templates of tasks that recur for lots of different stories so that I can reuse them	13





# Upcoming classes in Oslo

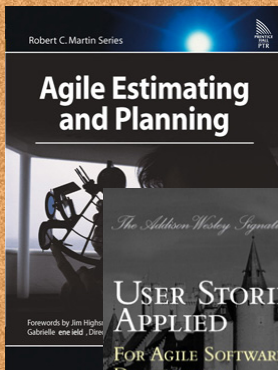
Date	What
22–23 June	Certified Scrum Product Owner
24–25 June	Certified ScrumMaster
12–14 October	Certified ScrumMaster (three-day)
15–16 October	Certified Scrum Product Owner
18–20 January	Certified ScrumMaster (three-day)
21–22 January	Certified Scrum Product Owner

Information and registration at  
[www.programutvikling.no](http://www.programutvikling.no)



© 2003–2009 Mountain Goat Software®

33



**Mike Cohn**  
[mike@mountaingoatsoftware.com](mailto:mike@mountaingoatsoftware.com)  
[www.mountaingoatsoftware.com](http://www.mountaingoatsoftware.com)  
 (720) 890-6110



© 2003–2009 Mountain Goat Software®

34