

# Agile Estimating and Planning

Mike Cohn

All slides © Mountain Goat Software

## My background



- Programming for 20 years
- Author of
  - *User Stories Applied*
  - *Agile Estimating and Planning* (mid 2005)
  - Books on Java, C++ and database programming
- Past consulting to Sun Microsystems, Nielsen Media Research, Microsoft, Yahoo, Webroot, many others
- Founding member and director of the Agile Alliance

All slides © Mountain Goat Software

## Today's agenda

---

- Why traditional approaches fail
- Agile planning
- Estimating work
- Estimating velocity
- Release planning
- Why agile planning works

All slides © Mountain Goat Software

## Why plans go wrong

---

1. Tasks are assumed to be independent

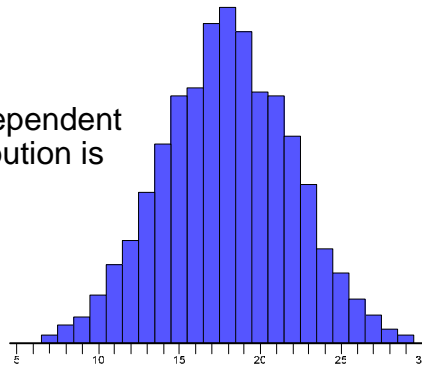
2. Lateness is passed down the schedule; earliness is not

3. The Student Syndrome

All slides © Mountain Goat Software

# 1. Task independence

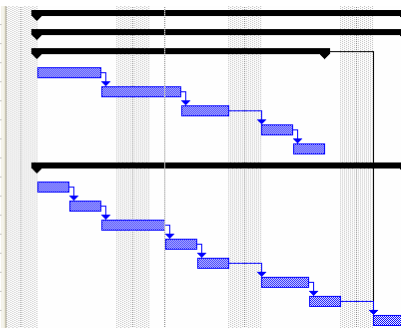
- Sum of five dice
- Central Limit Theorem
  - Sum of a number of independent samples from any distribution is approximately normally distributed
- This means that
  - some are bigger
  - some are small
  - but overall things average out



All slides © Mountain Goat Software

# Does CLT apply to software?

Sprint 4			
Analysis Manager	17 days	Mon 3/4/02	Tue 3/26/02
Database	14 days	Mon 3/4/02	Thu 3/21/02
Linkage	4 days	Mon 3/4/02	Thu 3/7/02
SIMWALK	3 days	Fri 3/8/02	Tue 3/12/02 4
CRI-Map	3 days	Wed 3/13/02	Fri 3/15/02 5
Genehunter	2 days	Mon 3/18/02	Tue 3/19/02 6
Inheritance Checking	2 days	Wed 3/20/02	Thu 3/21/02 7
Client	17 days	Mon 3/4/02	Tue 3/26/02
Create, rename, delete analysis	2 days	Mon 3/4/02	Tue 3/5/02
Edit settings	2 days	Wed 3/6/02	Thu 3/7/02 10
Sample selection	2 days	Fri 3/8/02	Mon 3/11/02 11
Marker selection	2 days	Tue 3/12/02	Wed 3/13/02 12
Variable Selection	2 days	Thu 3/14/02	Fri 3/15/02 13
Integrate with Task Manager	3 days	Mon 3/18/02	Wed 3/20/02 14
Run in background	2 days	Thu 3/21/02	Fri 3/22/02 15
Invoke analysis	3 days	Mon 3/25/02	Tue 3/26/02 3,16

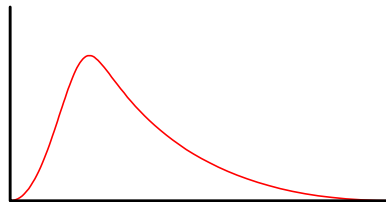


Highly correlated tasks

All slides © Mountain Goat Software

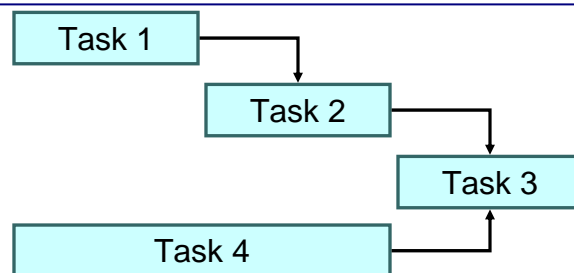
# CLT and software

- Tasks on a software Gantt chart are not independent
  - Many tasks involve similar work; if one estimate is wrong the others tend to be wrong
  - There may be systematic error in the estimates
- Software estimates tend not to be normally distributed
  - When asked for a point estimate programmers respond with the mode



All slides © Mountain Goat Software

## 2. Lateness is passed along the schedule



- Task 3 starts:
  - **LATE** if 1, 2 or 4 is late
  - **EARLY** only if 2 and 4 are early, and resource is available

All slides © Mountain Goat Software

### 3. Student syndrome

---

Definition

- Starting a task at the last possible moment that does not preclude an on-time completion

Example

- Starting a term paper the night before it's due

All slides © Mountain Goat Software

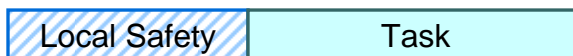
### What happens with student syndrome

---

- Estimate is based on this



- But we behave like this



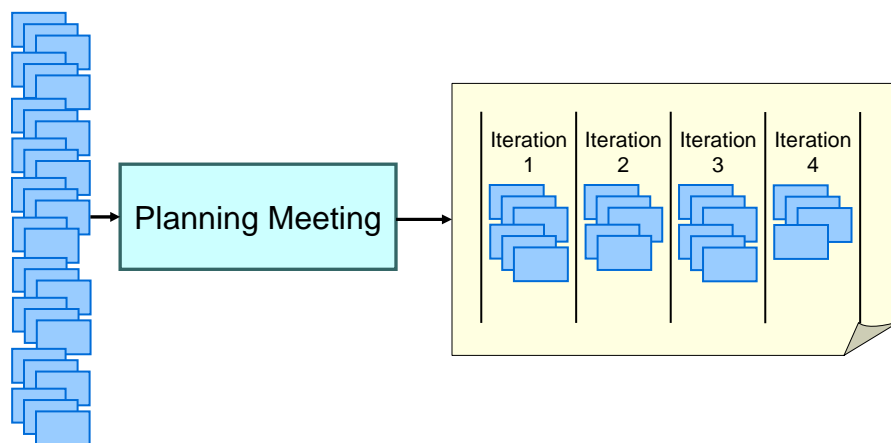
All slides © Mountain Goat Software

# Today's agenda

- ☑ Why traditional approaches fail
- ☐ Agile planning
- ☐ Estimating work
- ☐ Estimating velocity
- ☐ Release planning
- ☐ Why agile planning works

All slides © Mountain Goat Software

# Release planning



All slides © Mountain Goat Software

# Story points

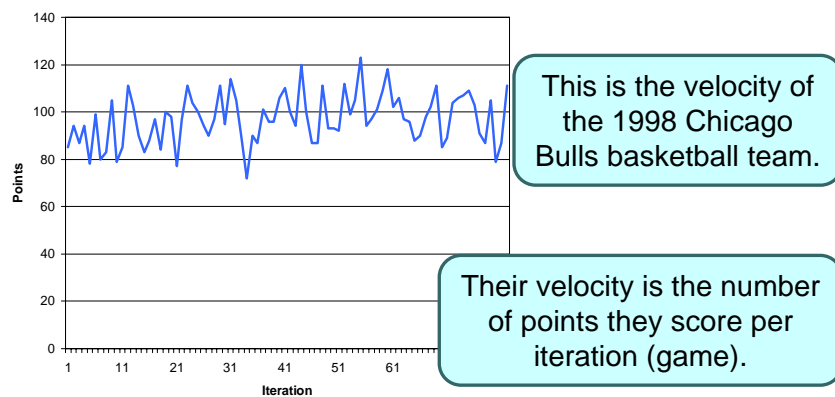
- Assume that each new feature (a “user story”) is assigned a number of points
  - Points represent how hard / time-consuming the feature will be to implement

Story A 3 points	Story B 5 points	Story C 4 points	Story D 5 points
Story E 2 points	Story F 5 points	Story G 3 points	Story H 1 point

All slides © Mountain Goat Software

# Velocity

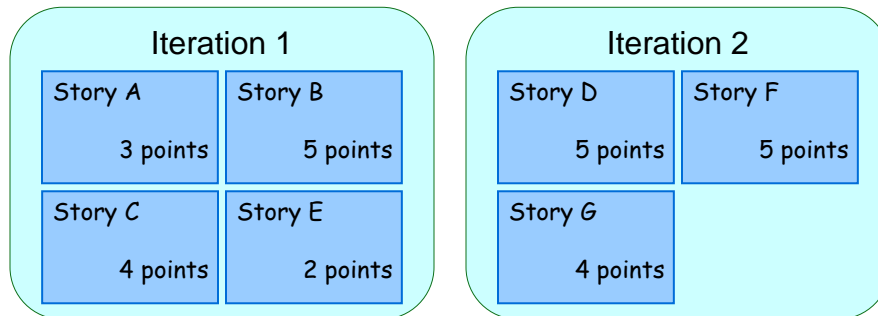
- Amount of work a team gets done per iteration



All slides © Mountain Goat Software

# Velocity

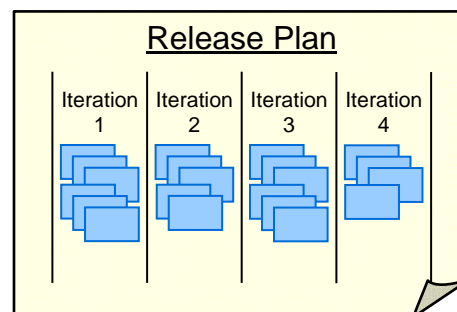
- We know the Bulls will score about 97 points per game
- If we knew how many points a software team would “score” per iteration we could assign work to iterations:



All slides © Mountain Goat Software

# The release plan

- We can create a release plan from:
  - the point estimate on each card
  - The velocity (“points per iteration”)
- Shows what will be worked on in each iteration
  - Each iteration is given the same number of points



All slides © Mountain Goat Software

## Where do we get the points?

---

- **Mostly we make them up!**
- Suppose we're sailing...
- ...we see this island through our telescope
- How far away is it?



All slides © Mountain Goat Software

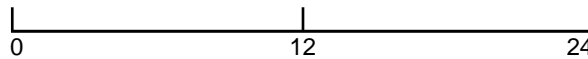
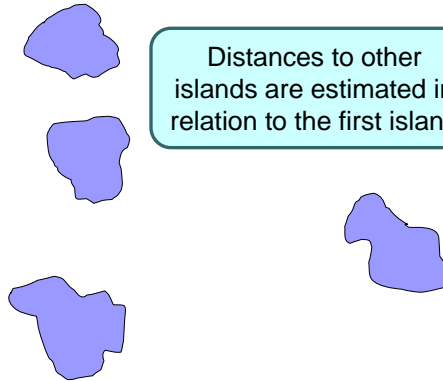
## How far away is the island?

---

- There's nothing else in sight so there's nothing to compare it to
  - Hard to guess then how many miles away it might be
- So, let's just say the distance is "12"

All slides © Mountain Goat Software

## Let's add more islands



All slides © Mountain Goat Software

## What do we know and not know?

We know...

- That three islands are about the same distance (12) away
- That another island is about twice as far away

We don't know...

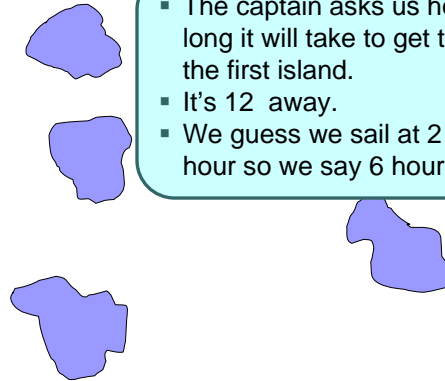
- how long it will take us to sail a distance of 12

But we know...

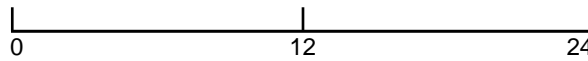
- That we can sail to all the islands 12 away in about the same time
- That it will take twice as long to sail to the island that is 24 away

All slides © Mountain Goat Software

## Estimating the first voyage

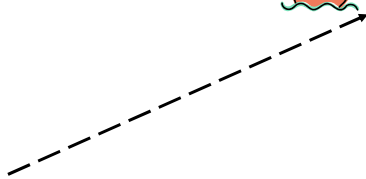


- The captain asks us how long it will take to get to the first island.
- It's 12 away.
- We guess we sail at 2 per hour so we say 6 hours.

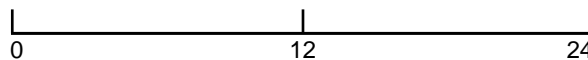


All slides © Mountain Goat Software

## We sail to the first island

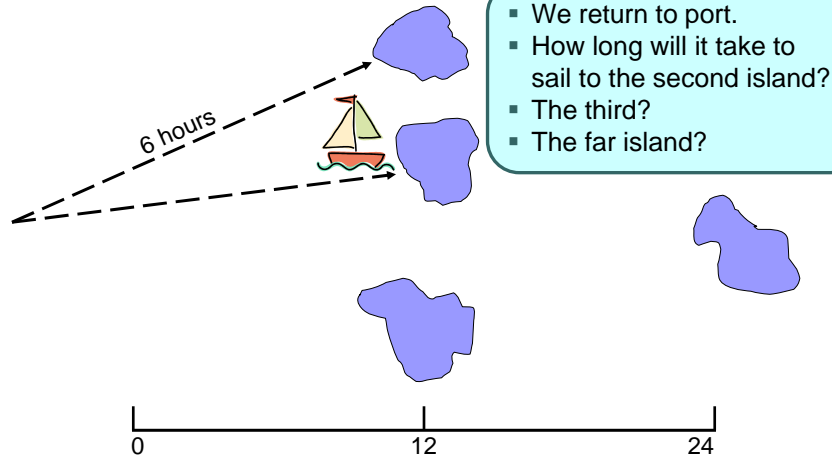


- Sailing to the first island takes 6 hours.
- We were right!
- We now know that we sail about 2 units per hour, or 12 units per six hours.



All slides © Mountain Goat Software

## How long will it take?



All slides © Mountain Goat Software

## No reliance on actual distances

We can now reliably estimate how long it will take to get to each of the four islands

Yet we still don't know how far away each island is

All slides © Mountain Goat Software

## What if we'd been wrong?



- What if we look through the telescope and still said 12...
- ...and estimate we can sail 6 per hour.
- We tell the captain we'll be there in 2 hours.

- The trip still takes 6 hours (not the 2 we told the captain)
- We now have a data point saying our rate of progress is 2 units per hour
- And we now know how long it will take to make all the other trips

software

## Relating this back to software

- The distances to the islands are our story points

Story A	Sail to Island A
3 points	12 points

- Both the boat and a development team have a semi-regular pace per period
  - The boat goes 12 units in 6 hours
  - A software team completes x story points per iteration

All slides © Mountain Goat Software

## Today's agenda

---

- Why traditional approaches fail
- Agile planning
- Estimating work
- Estimating velocity
- Release planning
- Why agile planning works

All slides © Mountain Goat Software

## Magnitude

---

- The general “bigness” of a task
- All that matters is the relative sizes
  - A 5-point story is half the size of a 10
  - A 20 is twice the size of a 10
- These are unitless *story points*
  - A 3,000 is twice the size of a 1,500

All slides © Mountain Goat Software

## Exercise



1) Assign “dog points” to each of the following types of dog.

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

All slides © Mountain Goat Software

## Ideal time

- How long something would take if
  - it's all you worked on
  - you had no interruptions
  - and everything you need was available
- The ideal time of a football game is 60 minutes
  - The elapsed time is much longer (3½ hours?)

All slides © Mountain Goat Software

## But, there's a problem

---

- Whose ideal time? Yours? Mine?

How do we add



All slides © Mountain Goat Software

## Archetypal programmer days

---

What?

- Define an archetypal programmer and estimate how long it will take her
- I like to use an “experienced senior programmer”
  - But it can vary and depends on the team

All slides © Mountain Goat Software

## Archetypal programmer days

---

Why?

- Estimates can be more honest
  - If questioned, “Oh, it wouldn’t take *me* that long.”
- Bias toward insufficient estimates goes away
- Estimates can be added and compared

All slides © Mountain Goat Software

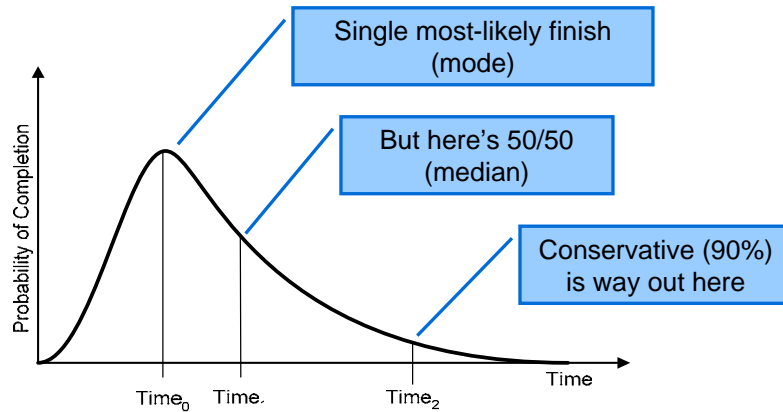
## Use the right units

---

- Can you distinguish a 17 from an 18?
  - Probably not
- Can you distinguish a  $\frac{1}{2}$  from a 1?
  - Probably
- Use units that make sense, such as:
  - 0,  $\frac{1}{2}$ , 1, 2, 3, 5, 10, 20, 40
  - 0,  $\frac{1}{2}$ , 1, 2, 3, 5, 8, 13, 21, 34

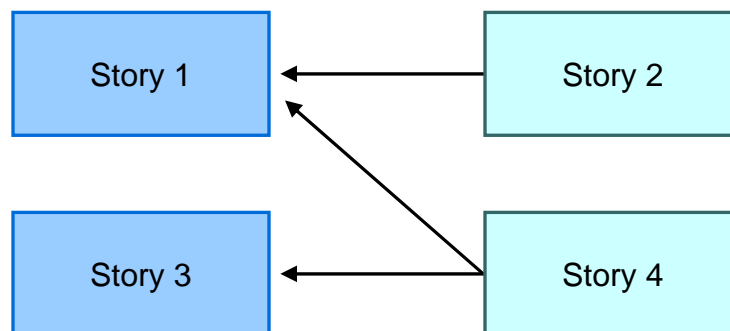
All slides © Mountain Goat Software

## State your assumptions



All slides © Mountain Goat Software

## Triangulate estimates



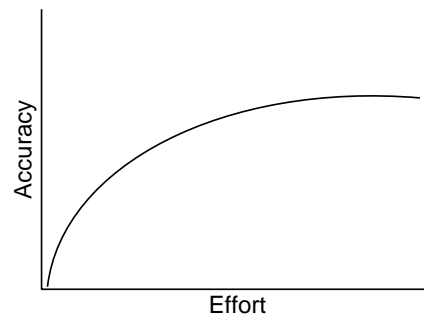
- Confirm a story to multiple other stories
- Group like-sized stories on table or whiteboard

All slides © Mountain Goat Software

## How much effort?

---

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



All slides © Mountain Goat Software

## Planning poker

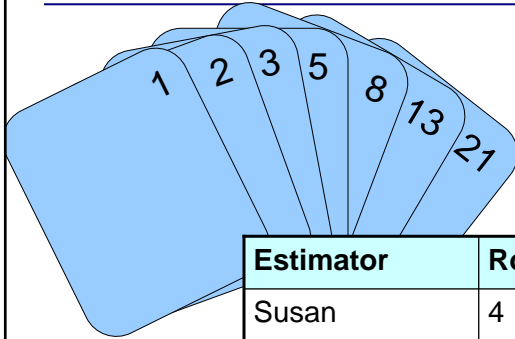
---

- An iterative approach to estimating
- Similar to wideband Delphi technique
- Steps
  1. Each estimator is given a deck of cards, each card has a valid estimate written on it
  2. The customer reads a story and it's discussed briefly
  3. Each estimator selects a card that's his estimate
  4. Cards are turned over so all can see them
  5. Discuss differences (especially outliers)
  6. Re-estimate until estimates converge

Reference: *Planning Poker* by James W. Grenning,  
[www.objectmentor.com](http://www.objectmentor.com)

All slides © Mountain Goat Software

## Planning poker—an example



Estimator	Round 1	Round 2
Susan	4	4
Vadim	7	5
Ann	2	4
Jay	4	4

All slides © Mountain Goat Software

## Today's agenda

- Why traditional approaches fail
- Agile planning
- Estimating work
- Estimating velocity
- Release planning
- Why agile planning works

All slides © Mountain Goat Software

## Estimating velocity

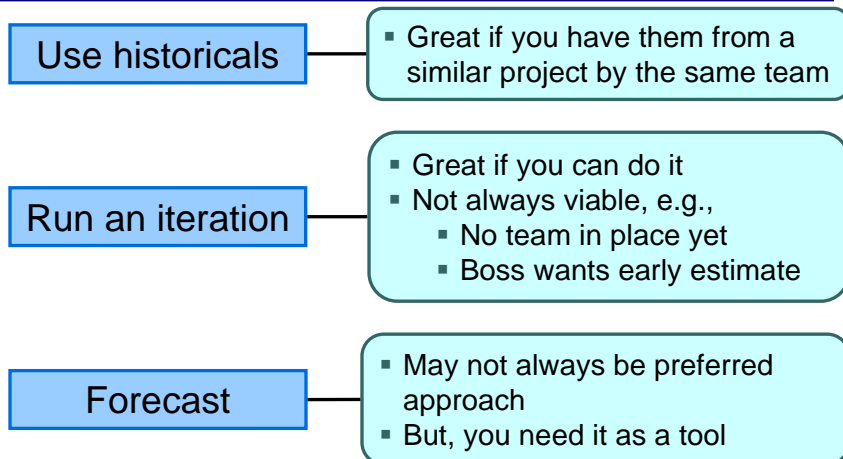
---

- How much can we do per iteration?
- Our best guess is that we can do next iteration what we did last iteration
  - “Yesterday’s Weather” (Beck & Fowler)
  - A martingale sequence

All slides © Mountain Goat Software

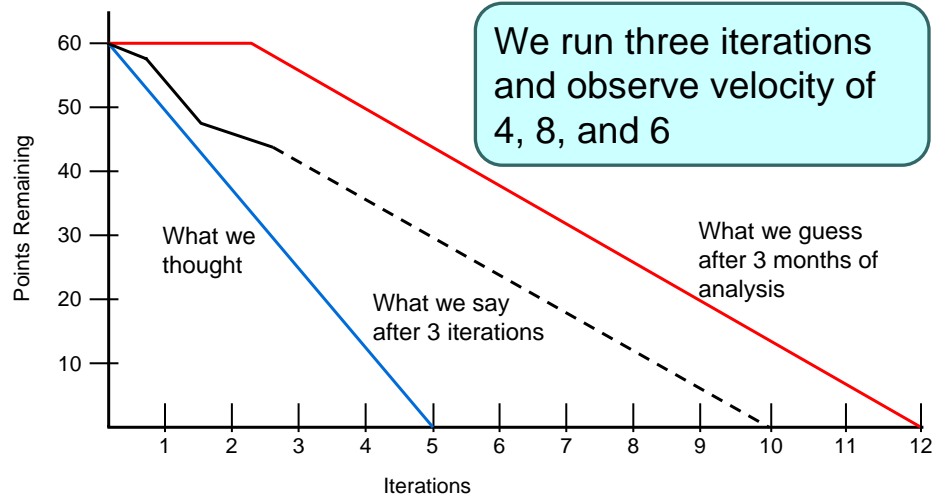
## Getting an initial velocity

---



All slides © Mountain Goat Software

## Committing to a plan



All slides © Mountain Goat Software

## Forecasting velocity from ideal time

- Estimate each developer's productivity relative to the archetypal programmer used in the estimates
- Considerations
  - Programming skill
  - Domain knowledge
  - Availability to actual code
  - Vacation

All slides © Mountain Goat Software

## Example: forecasting initial velocity

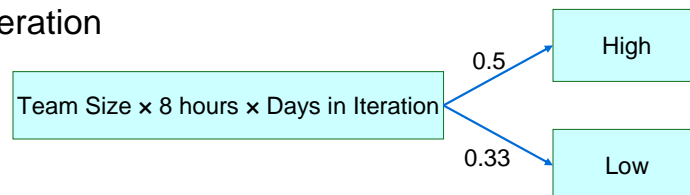
Developer	Iteration 1	Iteration 2	Iteration 3	Thereafter
Susan	.5	.5	.5	.5
Vadim	.2	.3	.4	.4
Ann		.2	.3	.4
Jay	.6	.7	.7	.7
Total	1.3	1.7	1.9	2.0

- This tells you how many ideal programmers you have working per calendar day
- If this team used 10-day iterations, they would have a velocity of 13 in the first iteration:  
$$1.3 * 10 = 13$$

All slides © Mountain Goat Software

## Forecasting velocity from magnitude

- Determine how many hours are available in the iteration



- Start with the highest-priority story
  - Disaggregate it into tasks of 1-16 hours each
  - See if it fits in the iteration
- Repeat with next highest-priority story
- Velocity = the points of the stories that fit

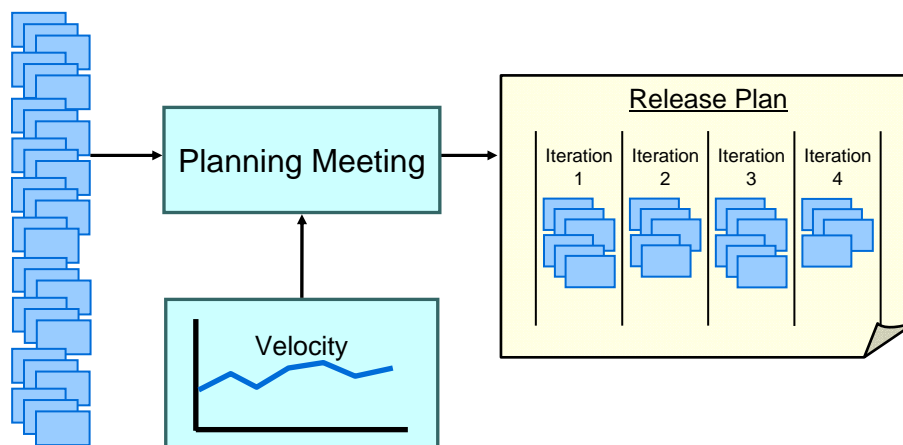
All slides © Mountain Goat Software

## Today's agenda

- ☑ Why traditional approaches fail
- ☑ Agile planning
- ☑ Estimating work
- ☑ Estimating velocity
- ☐ Release planning
- ☐ Why agile planning works

All slides © Mountain Goat Software

## Release planning



All slides © Mountain Goat Software

## An example with velocity = 14

Iteration 1

Iteration 2

Iteration 3

Story A	5	Story J	5
Story B	3	Story K	3
Story C	5	Story L	8
Story D	2	Story M	2
Story E	5		
Story F	2		
Story G	1		
Story H	8		
Story I	5		

All slides © Mountain Goat Software

## Today's agenda

- Why traditional approaches fail
- Agile planning
- Estimating work
- Estimating velocity
- Release planning
- Why agile planning works

All slides © Mountain Goat Software

# Why agile planning works

---

## ■ Why plans go wrong

1. Tasks are assumed to be independent

2. Lateness is passed down the schedule; earliness is not

3. The Student Syndrome

All slides © Mountain Goat Software

# Why agile planning works

---

1. Tasks are assumed to be independent

✓ Stories (the main unit of estimation) are largely independent.

All slides © Mountain Goat Software

## Why agile planning works

---

2. Lateness is passed down the schedule; earliness is not

- ✓ No overall Gantt or PERT chart
- ✓ Each day, each person picks what she'll do
  - ✓ Lateness doesn't pass down an agile plan
  - ✓ Earliness does pass down
- ✓ Naturally, there are some dependencies
  - ✓ But these are limited with an agile plan

All slides © Mountain Goat Software

## Why agile planning works

---

3. The Student Syndrome

- ✓ No Gantt chart saying what to do today and how long to take
- ✓ Increased visibility through daily standup meetings and pair programming

All slides © Mountain Goat Software

## Additionally

---

- ✓ Agile planning encourages and enforces continuous re-estimation and recalibration

All slides © Mountain Goat Software

## However...

---

“If you want a guarantee,  
buy a toaster.”

~Clint Eastwood



All slides © Mountain Goat Software

# My contact information



- [mike@mountaingoatsoftware.com](mailto:mike@mountaingoatsoftware.com)
- (303) 810-2190 (mobile)
- (720) 890-6110 (office)



- [www.mountaingoatsoftware.com](http://www.mountaingoatsoftware.com)
- [www.userstories.com](http://www.userstories.com)



All slides © Mountain Goat Software