Effective User Stories
for
Agile Software Development

Mike Cohn
September 20, 2004

My books and background

The Addison Wesley Signature Series

USER STORIES APPLIED
FOR AGILE SOFTWARE DEVELOPMENT

MIKE COHN
Foreword by Kent Beck

- Programming for 20 years
  - ☐ Author of four programming books
- Past consulting to Viacom, Fidelity Investments, Procter & Gamble, NBC, United Nations, Citibank, other smaller companies
- Founding member and director of the Agile Alliance
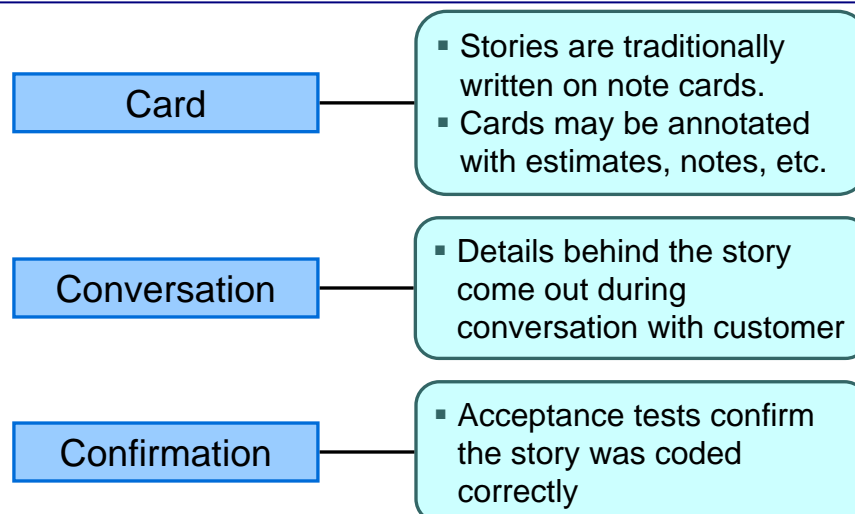- Currently VP, Engineering with Fast401k in Denver

2

1

Today's agenda

- What user stories are
- What user stories are not
- Why user stories?
- User role modeling
- INVEST in good stories
- Estimating
- Planning

3

Ron Jeffries' Three Cs

Card	▪ Stories are traditionally written on note cards. ▪ Cards may be annotated with estimates, notes, etc.
Conversation	▪ Details behind the story come out during conversation with customer
Confirmation	▪ Acceptance tests confirm the story was coded correctly

4

Samples – Travel Reservation System

A user can make a hotel reservation.

Users can see photos of the hotels.

A user can cancel a reservation.

Users can restrict searches so they only see hotels with available rooms.
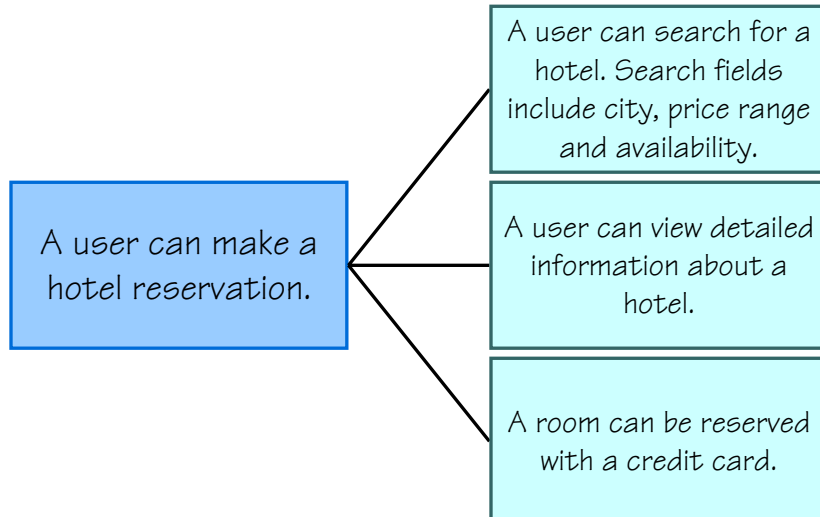
5

Where are the details?

- A user can make a hotel reservation.
  - □ Does she have to enter a credit card?
    - If so, what cards are accepted?
    - Is the charge applied immediately?
  - □ How can the user search for the hotel?
    - Can she search by city?
    - By quality rating?
    - By price range?
    - By type of room?
  - □ What information is shown for each room?
  - □ Can users make special requests, such as for a crib?

6

Details added in smaller "sub-stories"

```
A user can make a
hotel reservation.
```
→
```
A user can search for a
hotel. Search fields
include city, price range
and availability.
```
```
A user can view detailed
information about a
hotel.
```
```
A room can be reserved
with a credit card.
```

7

Details added as tests

- Tests are written on the back of a story card
  - Can be used to express additional details and expectations
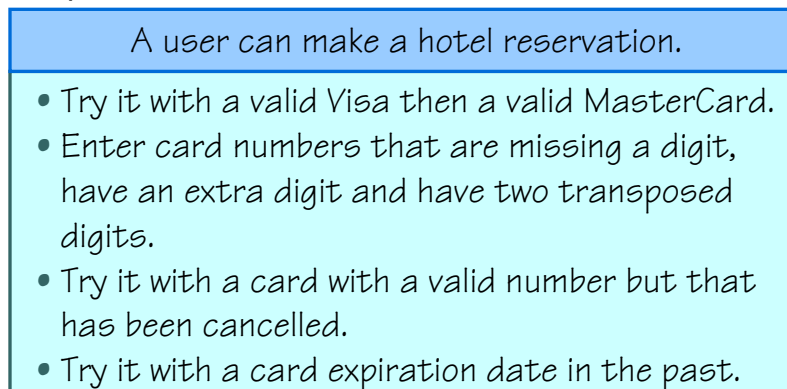
A user can make a hotel reservation.
• Try it with a valid Visa then a valid MasterCard.
• Enter card numbers that are missing a digit, have an extra digit and have two transposed digits.
• Try it with a card with a valid number but that has been cancelled.
• Try it with a card expiration date in the past.

8

4

Today's agenda

- ☑ What user stories are
- ☐ What user stories are not
- ☐ Why user stories?
- ☐ User role modeling
- ☐ INVEST in good stories
- ☐ Estimating
- ☐ Planning

9

User stories are not…

- ■ IEEE 830 Software Requirements Specifications
  - ☐ "The system shall…"
- ■ Use Cases
- ■ Scenarios
- ■ Features from FDD
  - ☐ "Calculate the total of a sale."
  - ☐ <action> the <result> <by|for|of|to> a(n) object

10

Stories are not IEEE 830

- An example IEEE 830 SRS:
    4. The system shall allow a room to be reserved with a credit card.
        1. The system shall accept Visa, MasterCard and American Express cards.
        2. The system shall charge the credit card the indicated rate for all nights of the stay before the reservation is confirmed.
    5. The system shall give the user a unique confirmation number

11

Problems with IEEE 830

- Time-consuming to write and read
- Tedious to read
    □ So readers skim or skip sections
- Assumes everything is knowable in advance

Feedback

Specs   Software
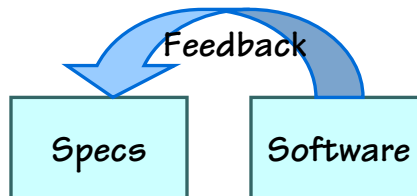
- Are these changes really a "change of scope"?

12

6

All requirements are not equal

- "Designers fix a top-level concept based on their initial understanding of a problem."[†]
- "May produce a solution for only the first few requirements they encounter."[‡]

Sources: [†]*Making Use* by John M. Carroll (2000) and [‡]*Technology and Change* by D.A. Schon (1967).

13

What are we building?

IEEE *Specs*

6. The product shall have a gas engine.
7. The product shall have four wheels.
   1. The product shall have a rubber tire mounted to each wheel.
8. The product shall have a steering wheel.
9. The product shall have a steel body.

Source: Adapted from *The Inmates are Running the Asylum* by Alan Cooper (1999).

14

What if we had stories instead?

The product makes it easy and fast for the user to mow her lawn.

The user is comfortable while using the product.

15

The product

16

Stories are not use cases

Title: Accept reservation for a room.
Primary Actor: Purchaser
…
Main Success Scenario:
1. Purchaser submits credit card number, date, and authentication information.
2. System validates credit card.
3. System charges credit card full amount for all nights of stay.
4. Purchaser is given a unique confirmation number.

17

Stories are not use cases

Extensions:
2a The card is not a type accepted by the system.
    2a1   System notifies the user to use a different card.
2b The card is expired.
    2b1   System notifies the user to use a different card.

3a The card has insufficient available credit.
    3a1   System charges as much as it can to the current card.
    3b1   User is told about the problem and asked to enter a second card; use case continues at 2

18

9

Differences between use cases and stories

- Scope
- Completeness
- Longevity
- Purpose
  - ☐ Use cases
    - Document agreement between customer and developers
  - ☐ Stories
    - Written to facilitate release and iteration planning
    - Placeholders for future conversations
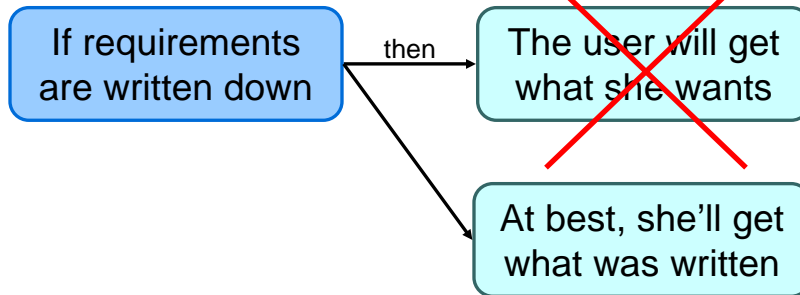
19

Today's agenda

- ☑ What user stories are
- ☑ What user stories are not
- ☐ Why user stories?
- ☐ User role modeling
- ☐ INVEST in good stories
- ☐ Estimating
- ☐ Planning

20

10

So, why user stories?

- Shift focus from writing to talking

If requirements are written down	→ then →	The user will get what she wants ✗
	→	At best, she'll get what was written

- "You built what I asked for, but it's not what I need."

21

Words are imprecise

Entrée comes with soup or salad and bread.

- (Soup or Salad) and Bread
- (Soup) or (Salad and Bread)

22

11

A panda walks into a restaurant…

Eats, shoots, and leaves

Eats shoots and leaves

23

Actual examples

The user can enter a name. It can be 127 characters.

- Must the user enter a name?
- Can it be other than 127 chars?

The system should prominently display a warning message whenever the user enters invalid data.

- What does *should* mean?
- What does *prominently display* mean?
- Is *invalid data* defined elsewhere?

24

Another real example

"I handed in a script last year and the studio didn't change one word."

"The word they didn't change was on page 87."

~Steve Martin

25

Words have multiple meanings

Buffalo buffalo buffalo.	— ▪ Bison intimidate bison.
Buffalo buffalo Buffalo buffalo.	— ▪ Bison intimidate bison from Buffalo.
Buffalo buffalo buffalo buffalo.	— ▪ Bison intimidated by bison intimidate bison. ▪ Bison from Buffalo intimidate bison.

26

13

Additional reasons

- Stories are comprehensible
  - Developers and customers understand them
  - People are better able to remember events if they are organized into stories[†]
- Stories are the right size for planning
- Support and encourage iterative development
  - Can easily start with epics and disaggregate closer to development time

[†]Bower, Black, and Turner. 1979. *Scripts in Memory for Text.*

27

Yet more reasons

- Stories support opportunistic development
  - We design solutions by moving opportunistically between top-down and bottom-up approaches[†]
- Stories support participatory design
  - Participatory design
    - The users of the system become part of the team designing the behavior of the system
  - Empirical design
    - Designers of the new system make decisions by studying prospective users in typical situations

[†]Guindon. 1990. *Designing the Design Process.*

28

14

Today's agenda

☑ What user stories are
☑ What user stories are not
☑ Why user stories?
❑ User role modeling
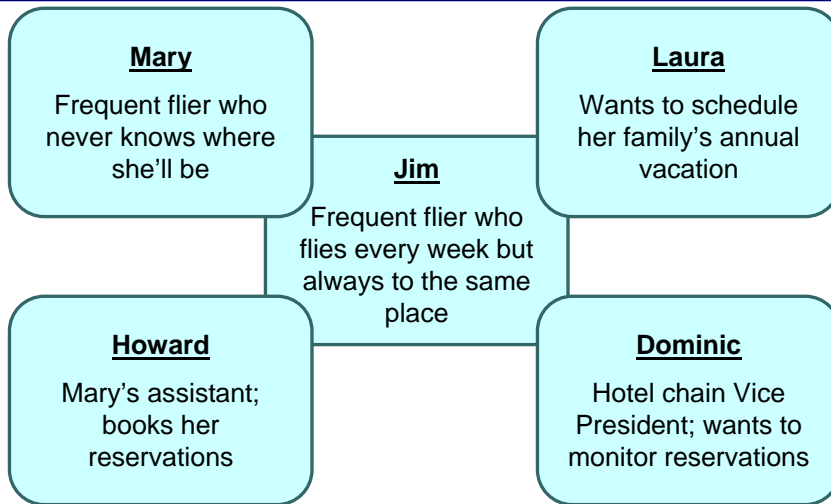❑ INVEST in good stories
❑ Estimating
❑ Planning

29

"The User"

■ Many projects mistakenly assume there's only one user:
  □ "The user"
■ Write all stories from one user's perspective
■ Assume all users have the same goals
■ Leads to missing stories

30

Travel Site—Who's the user?

Mary

Frequent flier who never knows where she'll be

Laura

Wants to schedule her family's annual vacation

Jim

Frequent flier who flies every week but always to the same place

Howard

Mary's assistant; books her reservations

Dominic

Hotel chain Vice President; wants to monitor reservations
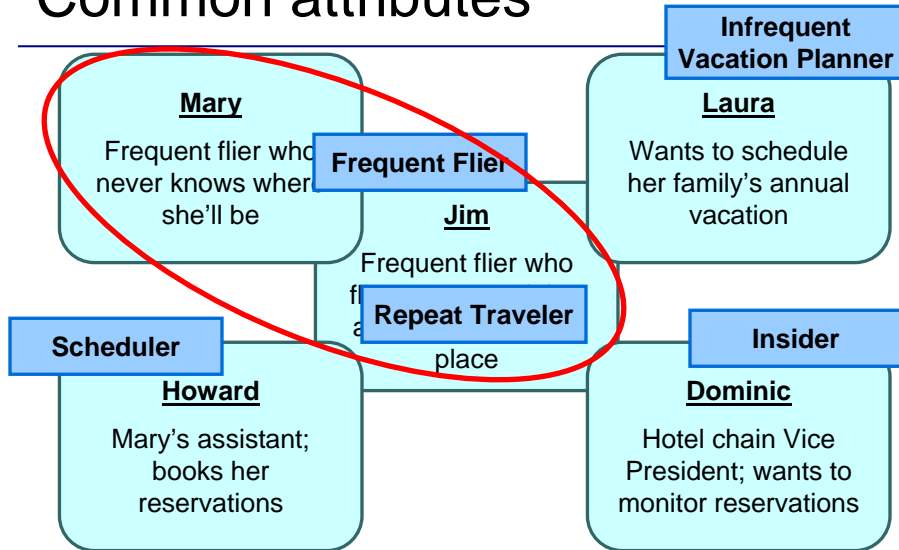
31

User roles

- Broaden the scope from looking at one user
- Allows users to vary by
  - What they use the software for
  - How they use the software
  - Background
  - Familiarity with the software / computers
- Used extensively in usage-centered design
- Definition
  - A user role is a collection of defining attributes that characterize a population of users and their intended interactions with the system.

Source: *Software for Use* by Constantine and Lockwood (1999).

32

16

Common attributes

Mary

Frequent flier who never knows where she'll be

Frequent Flier

Jim

Frequent flier who f... a... place

Scheduler

Howard

Mary's assistant; books her reservations

Repeat Traveler

Infrequent Vacation Planner

Laura

Wants to schedule her family's annual vacation

Insider

Dominic

Hotel chain Vice President; wants to monitor reservations

33

User role modeling

Identify attributes that distinguish one user role from another

- How often the software will be used
- Level of domain expertise
- General level of computer proficiency

- Level of proficiency with this software
- General goals for using the software

34

17
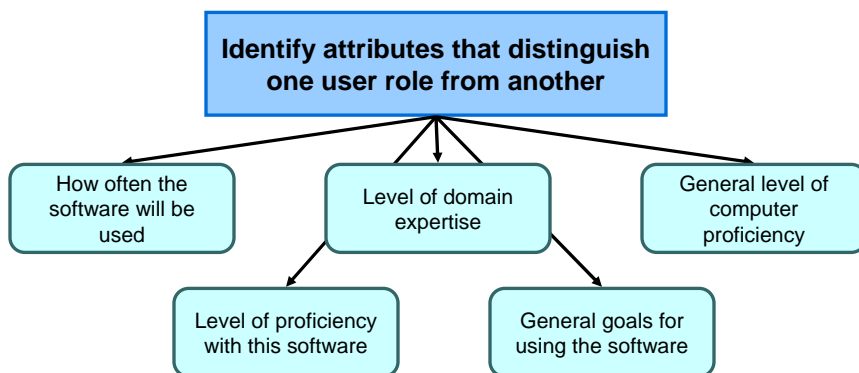
Document the user role

User Role: Infrequent Vacation Planner

Not particularly computer-savvy but quite adept at using the web. Will use the software infrequently but intensely (perhaps 5 hours to research and plan a trip). Values richness of experience (lots of content) over speed. But, software must be easy to learn and also easily recalled months later.

35

Advantages of using roles

Users become tangible	Start thinking of software as solving needs of real people.
Avoid saying "the user"	Instead we talk about "a frequent flier" or "a repeat traveler"
Incorporate roles into stories	"As a <role>, I want <story> so that <benefit>."

36

Exercise

We have been asked to develop a new online dating website.

1) What roles are there?
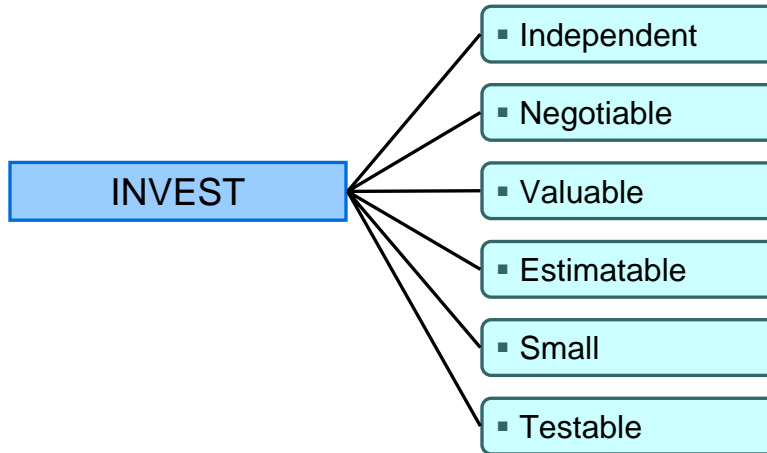2) Which roles are the most important to satisfy?

Today's agenda

- ☑ What user stories are
- ☑ What user stories are not
- ☑ Why user stories?
- ☑ User role modeling
- ❑ INVEST in good stories
- ❑ Estimating
- ❑ Planning

What makes a good story?

INVEST
- Independent
- Negotiable
- Valuable
- Estimatable
- Small
- Testable

Thanks to Bill Wake for the acronym.
See www.xp123.com.

39

Independent

- **Avoid introducing dependencies**
  - Leads to difficulty prioritizing and planning

A company can pay for a job posting with a Visa card.

?

A company can pay for a job posting with an AmEx card.

?

A company can pay for a job posting with a MasterCard.

?

- The first of these stories will take 3 days to develop
  - It doesn't matter which is first
- The others will take 1 day

40

20

Making stories independent

| Combine the stories | • A customer can pay with a credit card. |

| Split across a different dimension | • A customer can pay with one type of credit card.
• A customer can pay with two other types of credit cards. |

| Write two estimates and move on | • 3 days if first; 1 otherwise |

Negotiable

- Stories are not
  - □ Written contracts
  - □ Requirements the software must fulfill
- Do not need to include all details
- Too many details give the impressions of
  - □ false precision or completeness
  - □ that there's no need to talk further
- Need some flexibility so that we can adjust how much of the story gets implemented
  - □ If the card is contract then it needs to be estimated like a contract

Is this story negotiable?

A company can pay for a job posting with a credit card.

Note: Accept Visa, MasterCard, and American Express. Consider Discover. On purchases over $100, ask for card ID number from back of card. The system can tell what type of card it is from the first two digits of the card number. The system can store a card number for future use. Collect the expiration month and date of the card.

How about this one?

A company can pay for a job posting with a credit card.

Note: Will we accept Discover cards?
Note for UI: Don't have a field for card type (it can be derived from first two digits on the card).

Valuable

- Stories must be valuable to either:

Users

- A user can search for a job by title and salary range.

Purchasers

- Throughout the project, the development team will produce documentation suitable for an ISO 9001 audit.
- The development team will produce the software in accordance with CMM level 3.
- All configuration information is read from a central location.

45

Stories valued by developers

- Should be rewritten to show the benefit

All connections to the database are through a connection pool.	→	Up to 50 users should be able to use the application with a five-user database license.
All error handling and logging is done through a set of common classes.	→	All errors are presented to the user and logged in a consistent manner.

46

23

Estimatable

- Because stories are used in planning
- A story may not be estimatable if:

Developers lack domain knowledge	▪ New users are given a diabetic screening.
Developers lack technical knowledge	▪ A user can select to see all text on the site in a larger font.
The story is too big	▪ A user can find a job.

Small

- Large stories (epics) are
  - □ hard to estimate
  - □ hard to plan
    - ▪ They don't fit well into single iterations
- Compound story
  - □ An epic that comprises multiple shorter stories
- Complex story
  - □ A story that is inherently large and cannot easily be disaggregated into constituent stories

Compound stories

- Often hide a great number of assumptions

A user can post her resume.

- A resume includes separate sections for education, prior jobs, salary history, publications, etc.
- Users can mark resumes as inactive
- Users can have multiple resumes
- Users can edit resumes
- Users can delete resumes

49

Splitting a compound story

Split along operational boundaries (CRUD)

- A user can create resumes, which include education, prior jobs, salary history, publications, presentations, community service, and an objective.
- A user can edit a resume.
- A user can delete a resume.
- A user can have multiple resumes.
- A user can activate and inactivate resumes.

50

Splitting a compound story, cont.

Split along data boundaries

- A user can add and edit educational information on a resume.
- A user can add and edit prior jobs on a resume.
- A user can add and edit salary history on a resume.
- A user can delete a resume.
- A user can have multiple resumes.
- A user can activate and inactivate resumes.

51

Testable

- Tests demonstrate that a story meets the customer's expectations
- Strive for 90+% automation

A user must find the software easy to use. → A novice user is able to complete common workflows without training.

A user must never have to wait long for a screen to appear. → New screens appear within 2 seconds in 95% of all cases.

52

26

Exercise

1) Write some stories, based on the user roles, for our online dating website.

Tip: try this template:

"As a <role>, I want to <story> so that <benefit>."

53

Additional guidelines for good stories

- Start with goals
- Slice the cake
- Write closed stories
- Put constraints on cards
- Size the story to the horizon

- Keep the UI out as long as possible
- Some things aren't stories
- Include user roles in the stories
- Write for one user
- Write in active voice

Don't forget the purpose

54

Today's agenda

☑ What user stories are

☑ What user stories are not

☑ Why user stories?

☑ User role modeling

☑ INVEST in good stories

☐ **Estimating**

☐ Planning

55

Two approaches

Ideal Time

Magnitude

56

Ideal time

- An estimate of how long something would take if:
  - It's the only thing you work on
  - You have everything you need at hand when you start
  - There are no interruptions

57

Elapsed time vs. ideal time

| Elapsed time | - Monday has 8 hours
- Each week has 40 hours |

| Ideal time | - Time on task
- Monday has
  - 3 hours of meetings
  - 1 hour of email
  - 4 hours of programming (time-on-task) |

58

29

"How long will this take?"

- "Two weeks."
- Two *calendar* weeks or two weeks worth of *time on task*?

June 04						
Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
May 31	June 1	2	3	4	5	6
TODAY						
7	8	9	10	11	12	13
				▼		
14	15	16	17	18	19	20
21	22	23	24	25	26	27
	●					
28	29	30	July 1	2	3	4

59

Factors affecting ideal time

- Vacations
- Sick time
- All-company meetings
- Department meetings
- Demos
- Debugging

- Personnel issues
- Phone calls
- Special projects
- Training
- Email
- Sabbaticals

- Reviews & walk-throughs
- Interviewing candidates
- Spikes
- Leaves of absence
- Talking to vendors

60

30

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
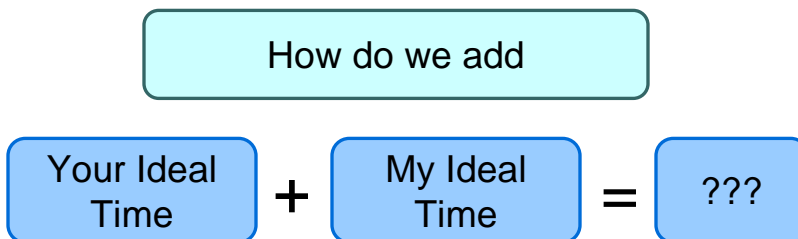
But, there's a problem

- Whose ideal time? Yours? Mine?

How do we add

Your Ideal Time **+** My Ideal Time **=** ???

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

63

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

64

Disadvantages of ideal time

- Can't add your ideal time to my ideal time
  - □ Without estimating in something like "Archetypal Programmer" days
  - □ But it can be hard to estimate someone else's ideal time
- Need to re-estimate whenever we get better or when we know something new about a task
- Developers may make an implicit conversion
  - □ "Two ideal days is about a week. I think I could do this in a week. I'll say it's two ideal days."

65

Advantages of ideal time

- Very tangible and understandable
  - □ Easy to get started with
- Straightforward to convert from ideal time to calendar time

66

33

Magnitude

- The "bigness" of a task
- Influenced by
  - Complexity
  - Our current knowledge
  - How much of it there is
- Relative values are what is important:

> - "A login screen is a 2."
> - "A search feature is an 8."

> - "A login screen is small."
> - "A search feature is large."

67

What are the magnitudes of these?

Develop 100 screens, each with 2 fields

Code 1 screen with 200 fields on it

Remove the recursion from the ABC class and make it thread safe

Write a "Hello, World" servlet

68

Problems with magnitude

- Values must be meaningful and distinguishable
  - How do you tell a "67" from a "68"?
- Eventually you need to convert an estimate of magnitude into an estimate of duration
  - "We'll be done in 8 mediums, 3 smalls and 4 larges."
  - "We'll be done in 43 Gummi Bears."
- Developers may make an implicit conversion
  - "Most 3s take a day, this seems like a day; I'll say it's a 3."
- Can feel very uncomfortable at first
- Very hard to estimate initial velocity

Advantages to magnitude

- Some developers find it much easier to say "this is like that"
- The abstractness can help developers from feeling coerced into giving an estimate that meets an expected deadline
  - "My boss wants this in two weeks, I guess I'll say 'two weeks.'"
- Can be done very quickly, once it's familiar
- Less need to re-estimate than ideal time
  - Something that used to take 1 ideal day might now take ½ ideal day (as the team improves)
  - Something that is "big" is still big; even though the team may be faster

Story points

- A story point is either:
  - ☐ 1 ideal day
  - ☐ 1 unit of measure for magnitude

71

What I do

Initially…
- Start with ideal time
- Gives the team a nice foundation for the initial stories
- Helps them get started
- I define "1 Story Point = 1 Ideal Day"

Then…
- Gradually convert team to thinking more about magnitude
- This story is like that story
- Stop talking about how long it will take

72

36

Use the right units

- Can you distinguish a 17-hour task from an 18-hour task?
- Can you distinguish a ½ day from a 1 day task?

> - Use units that make sense, such as:
>   - 0, ½, 1, 2, 3, 5, 10, 20, 40
>   - 0, ½, 1, 2, 3, 5, 8, 13, 21

73

The estimation meeting

- Bring the whole team (if possible & practical)
  - □ Programmers, testers, DBAs, etc.
- Invite the customer
  - □ Customer(s) participate in discussion but do not estimate directly
- Give estimate cards to estimators
  - □ Can be pre-printed or blank

74

Repeat for each story

1. A moderator reads a story and it's discussed briefly
2. Each estimator selects a card that is her estimate
3. Cards are turned over so all can see them
4. Discuss differences (especially outliers)
5. Re-estimate until estimates converge

75

An example

Estimator	Round 1	Round 2
Susan	4	4
Rafe	7	5
Ann	2	4
Sherri	4	4

76

How much effort?

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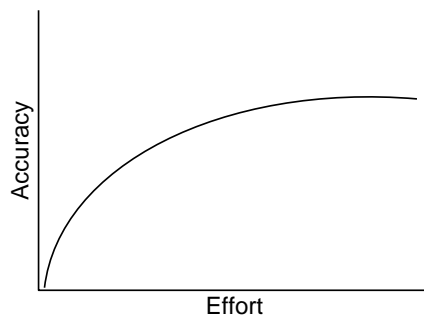
Exercise

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

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

Today's agenda

☑ What user stories are

☑ What user stories are not

☑ Why user stories?

☑ User role modeling

☑ INVEST in good stories

☑ Estimating

❑ Planning

What we'd like to do

- Take a prioritized stack of user stories
- Figure out how much we can do per iteration
- And then know how many iterations it will take

Iteration 1

Iteration 2

Iteration 3

Iteration 4
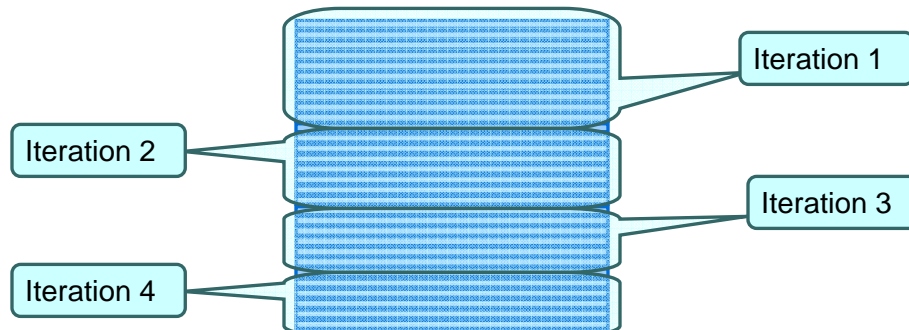
Different dimensions to prioritization

Technical

- Risk that the story cannot be completed as desired
- Impact the story will have on other stories if deferred

Customers / Users

- Desirability of the story to a broad base of users
- Desirability of the story to a small number of important users
- Cohesiveness of the story to other stories.

81

Who wins

- Customer wins—always
- But need developer input in order to prioritize

Customer cannot prioritize without knowing the cost of the stories

The user can book a new trip based on a previous trip.

3—5 days

Developers are best at identifying dependencies between stories

82

41

Split stories with mixed priorities

Users can search for magazine articles by author, publication name, title, date, or any combination of these.

Users can search for magazine articles by author and/or title.

Users can search for magazine articles by publication name, date or any combination of these.

Risky stories vs. juicy stories

- Agile is firmly in the camp of doing the "juicy bits" first
- But cannot totally ignore risk
  - If some stories are very risky, the developers need to tell the customer

Infrastructural stories

- Infrastructural stories are usually best assessed by the risk of deferring them (but still doing them later)

Be able to generate 50 stock chart images per second.

Is this performance achievable on targeted hardware?

Can we still use Java or should we do this natively?

What type of caching do we need to achieve this?

How much can we do per iteration?

- Velocity
- Our best guess is that we can do next iteration what we did last iteration
  - □ "Yesterday's Weather" (Beck & Fowler)
- But sometimes we don't have a last iteration

Getting an initial velocity

Use historicals
- Great if you have them from a similar project by the same team

Run an iteration
- Great if you can do it
- Not always viable, e.g.,
  - No team in place yet
  - Boss wants early estimate

Forecast
- May not always be preferred approach
- But, you need it as a tool

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

Example: forecasting initial velocity

Developer	Iteration 1	Iteration 2	Iteration 3	Thereafter
Susan	.5	.6	.7	.7
Ann	.5	.5	.5	.5
Randy	.2	.3	.4	.4
Clark		.2	.3	.4
Vlade	.5	.6	.7	.7
Chris	.8	.9	1.0	1.0
Total	2.5	3.1	3.6	3.7

- This tells you how many "archetypal programmers" you have working per calendar day

Forecasting velocity from magnitude

- Starting with the highest-priority story, select as many stories as you think will fit in the first iteration
  - Break each story into smaller tasks (< 1 calendar day)
  - When the iteration feels full, stop and see how many story points were brought in
  - That's your guess at velocity

What we can do

- Take a prioritized stack of user stories
- Grab an iteration's worth of points
- Keep grabbing until out of stories or time
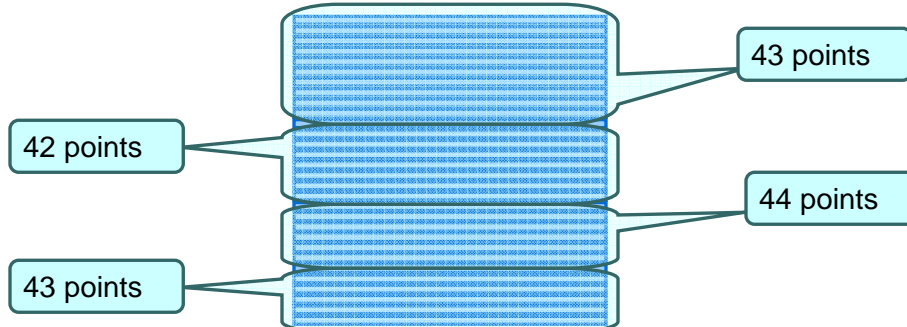
43 points

42 points

44 points

43 points

91

Today's agenda

☑ What user stories are
☑ What user stories are not
☑ Why user stories?
☑ User role modeling
☑ INVEST in good stories
☑ Estimating
❑ Planning
   ❑ Planning with a project buffer

92

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 |

What happens with student syndrome

- Estimate is based on this

| Task | Local Safety |

- But we behave like this

| Local Safety | Task |

My trip to the airport

Find Keys — 1 5

Drive to Airport — 45 30

Park — 5 10

Check in — 7 30

Security — 7 30

| 50% Estimate | Time = 1:05 |
| Buffer to 90% | Time = 1:45 |

Total = 2:50 minutes
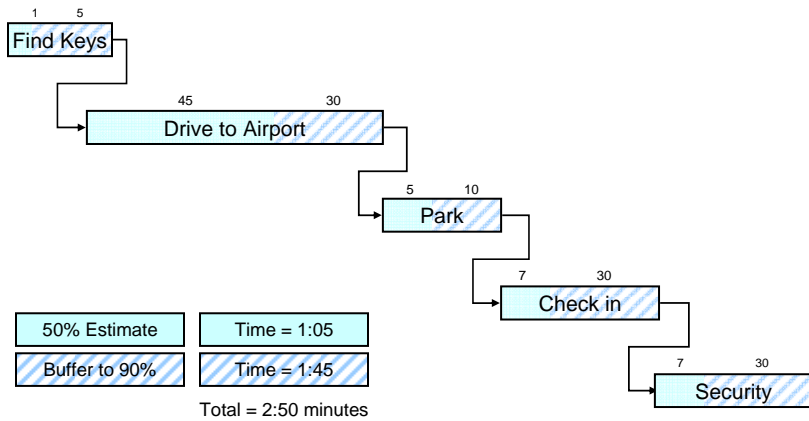
Distribution of completion times

Single most-likely finish; what many developers offer

But here's 50/50

Conservative (90%) is way out here

Probability of Completion

Time

$Time_0$ $Time_1$ $Time_2$

48
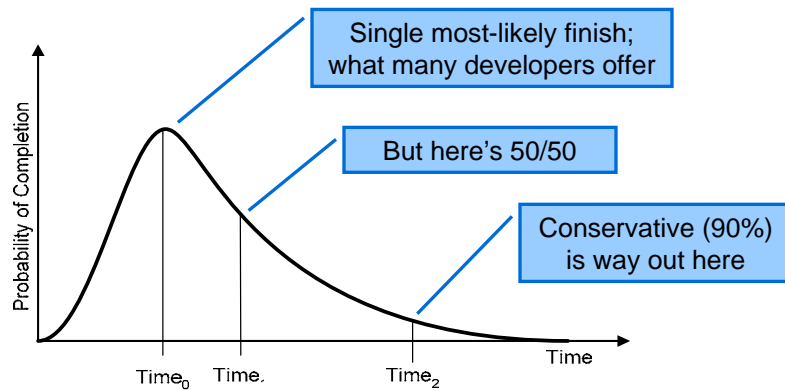
Give both 50% and 90% estimates

- 50% estimates
  - Remove all *local safety*: no "padding"
  - An estimate you should / will miss more than half the time
- 90% estimates
  - Not really a worst case
    - No lightning strikes or busses running over people
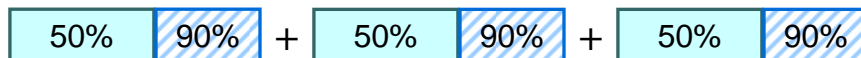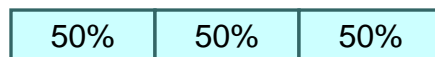 - Keep in mind that you'll even exceed this estimate occasionally

Release planning

- We can't add the 50% estimates together
  - That assumes everything goes smoothly
  - Overall schedule will be too short

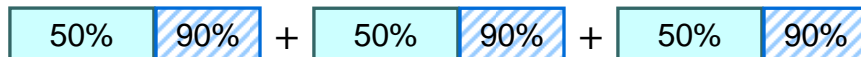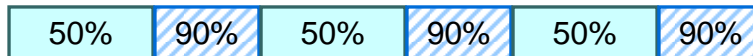| 50%	90%	+	50%	90%	+	50%	90%

!=

50%	50%	50%

49

Release planning

- We can't add the 90% estimates together
  - That assumes that everything goes wrong
  - Overall schedule will be too long

| 50% | 90% | + | 50% | 90% | + | 50% | 90% |

!=

| 50% | 90% | 50% | 90% | 50% | 90% |

99

The solution

- We add the 50% estimates
- And buffer the overall project, rather than the tasks

| 50% | 90% | + | 50% | 90% | + | 50% | 90% |

==

| 50% | 50% | 50% | 90% |

100

50
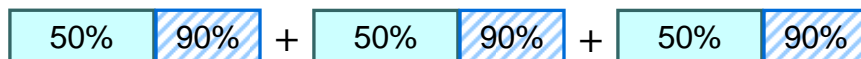
My airport trip with a project buffer

1	
45	
5	
7	
7	
53	

50% Estimate	Time = 1:05
Buffer	Time = 0:53

Total = 1:58

Was 2:50
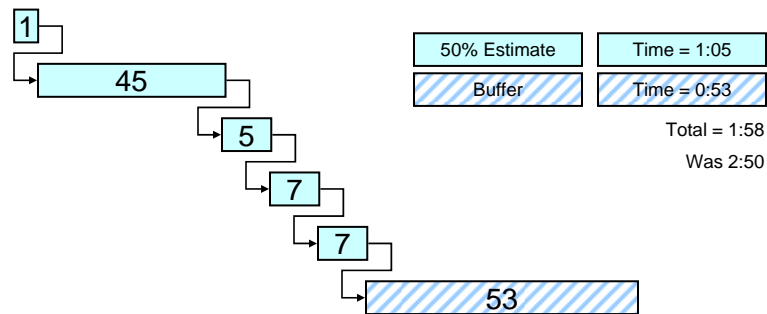
A project buffer isn't padding

- Padding is extra time you don't think you'll need but add to be safe
- You *will* need the project buffer
  - Even with the project buffer you're not guaranteed to be done on time
- I had a 3% chance of making it to my flight in 65 minutes

$$50\% \times 50\% \times 50\% \times 50\% \times 50\% = 3.125\%$$

1:05	53

- Would you call something that increases your odds of success from 3% "padding"?

51

How long should the buffer be?

- Simple rule
  - Use 50% of the unbuffered (50%) schedule
- More sophisticated, usually better

$$\sqrt{(w_1 - a_1)^2 + (w_2 - a_2)^2 + \cdots + (w_n - a_n)^2}$$

  - w = worst case
  - a = average case

103

Sample buffer calculation

Story	50%	90%	(90%—50%)²
Story 1	2	5	9
Story 2	3	5	4
Story 3	1	1	0
Story 4	1	3	4
Story 5	5	8	9
Story 6	5	6	1
Total	17	28	27

$$Schedule = 17 + \sqrt{27} = 17 + 5.2 = 22$$

104

52

Full example of planning a release

Story	50%	90%	(90%—50%)²
Story 1	2	5	9
Story 2	3	5	4
…	…	…	0
Total	**117**	**200**	**1089**

$$117 + \sqrt{1089} = 117 + 33 = 150$$

Developer	Iteration 1	Iteration 2	Iteration 3	Thereafter
Susan	.5	.6	.7	.7
Ann	.5	.5	.5	.5
Randy	.2	.3	.4	.4
Clark		.2	.3	.4
Vlade	.5	.6	.7	.7
Chris	.8	.9	1.0	1.0
Total	**2.5**	**3.1**	**3.6**	**3.7**

Example, continued

Velocity estimates from previous slide

Iteration	Duration (Days)	Daily Velocity	Story Points in iteration	Cumulative Story Points
Iteration 1	10	2.5	25	25
Iteration 2	10	3.1	31	56
Iteration 3	9	3.6	32	88
Iteration 4	10	3.7	37	125
Iteration 5	10	3.7	37	162

Company holiday

Accumulate 150 Story Points sometime during Iteration 5

For more on user stories

USER STORIES APPLIED
FOR AGILE SOFTWARE DEVELOPMENT
MIKE COHN
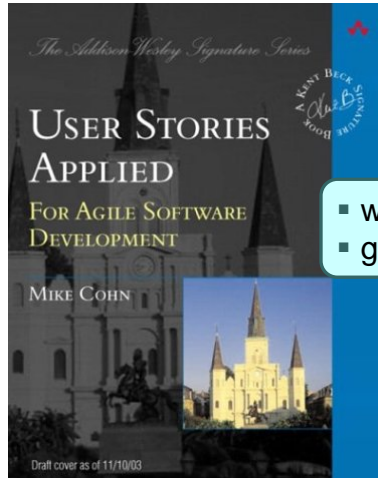
- www.userstories.com
- groups.yahoo.com/group/userstories

107

Where to go next?

Agile Alliance

Agile Planning
- groups.yahoo.com/agileplanning
- www.mountaingoatsoftware.com/agileplanning

Agile in General
- www.agilealliance.com

Scrum
- www.mountaingoatsoftware.com/scrum

108

55

My contact information

Email

- mike@userstories.com
- mike.cohn@computer.org

- www.mountaingoatsoftware.com
- www.userstories.com

Websites