CitiusTech

Smart Scrum

Session 3

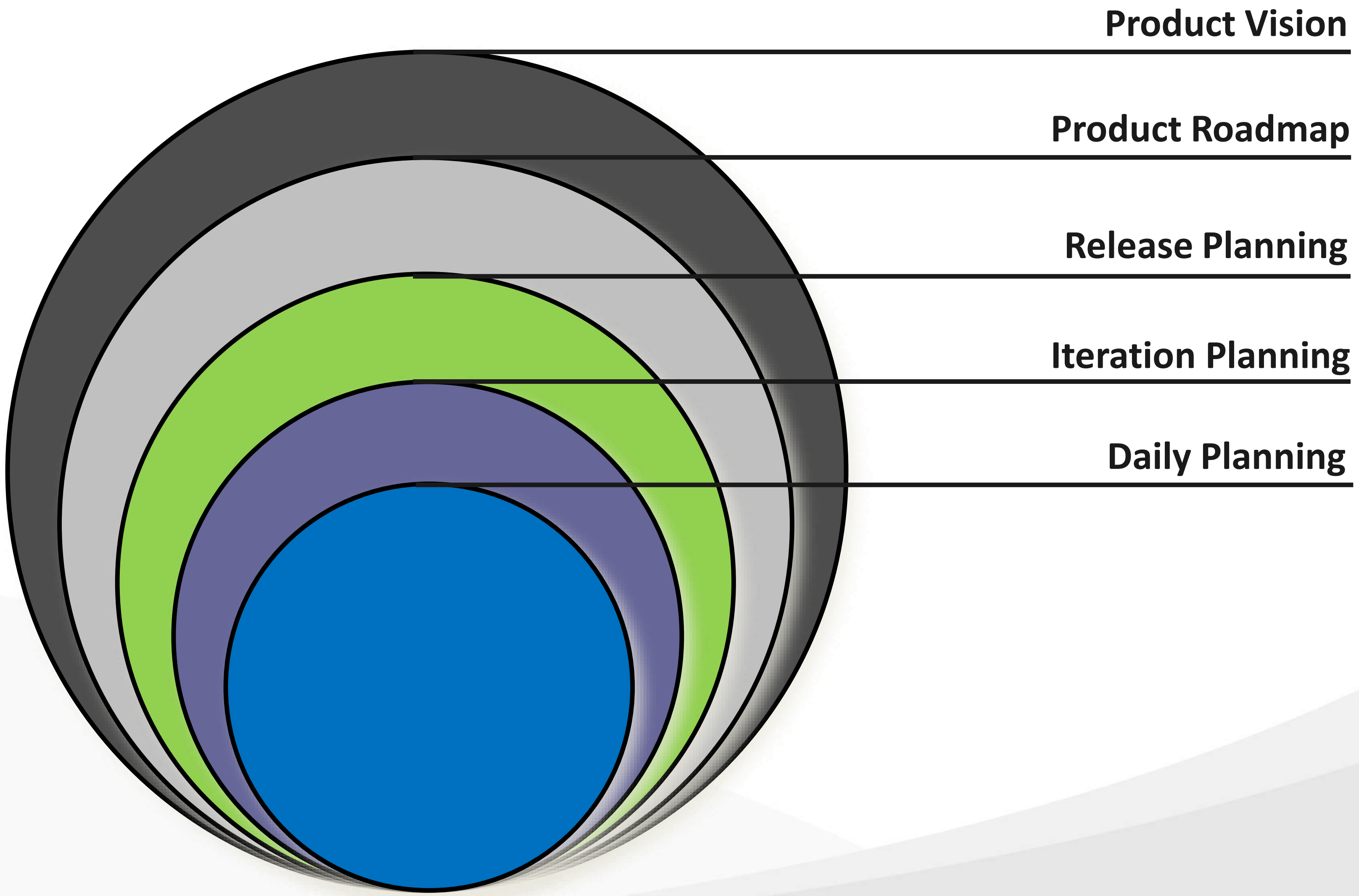
Version 1.0

December 2019

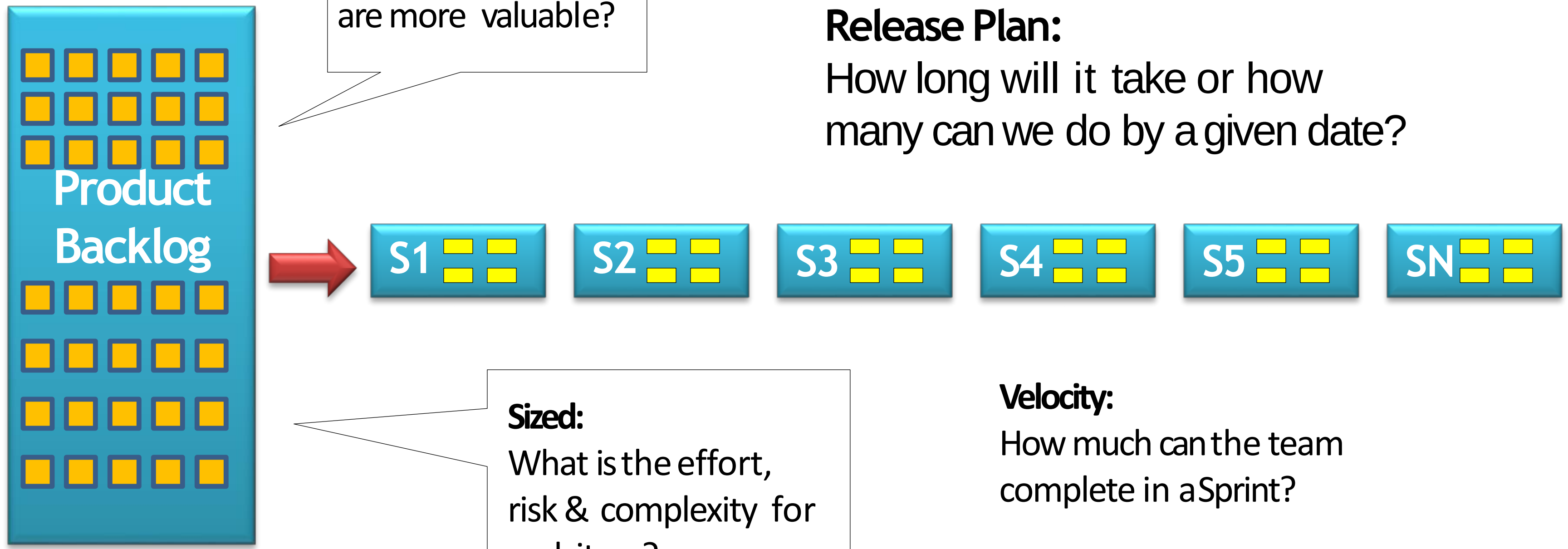
Agenda

Levels of Agile Planning

- Agile team can plan their projects at below mentioned levels-



The Elements of Agile Planning



Priorities:
Which items are more valuable?

Release Plan:
How long will it take or how many can we do by a given date?

Sized:
What is the effort, risk & complexity for each item?

Velocity:
How much can the team complete in a Sprint?

Value:
What capabilities are needed for financial success?

CitiusTech

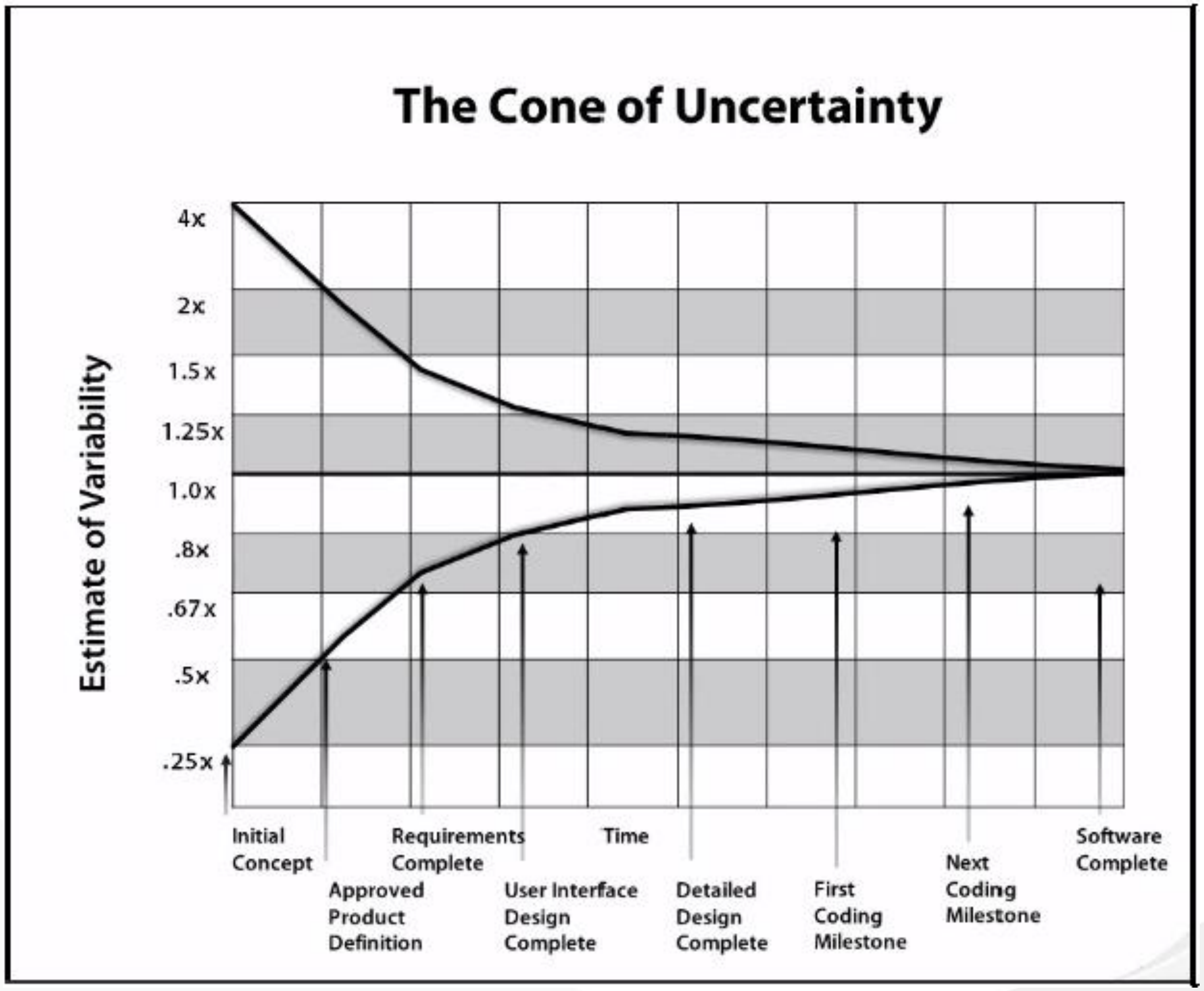
Introduction to Estimation

- What is Estimation?
  - *“Estimation is the calculated approximation of a result which is usable even if input data may be incomplete or uncertain” - Wikipedia*
- Why do we Estimate?
- Why Estimation fails?

Why Do We Estimate?

- To plan of the future
- To schedule the tasks/work at each stage
- To avoid running behind the schedule situations
- What should be considered while estimation?
- Reminds us that our estimates are guesses
- Acknowledges the inherent complexities and uncertainty that comes with writing software
- Buffer time
- Leave plans
- Correct way of measure for tasks
- Estimations will not be perfect and will fail

Why Estimation Fails? (1/4)



Why Estimation Fails? (2/4)

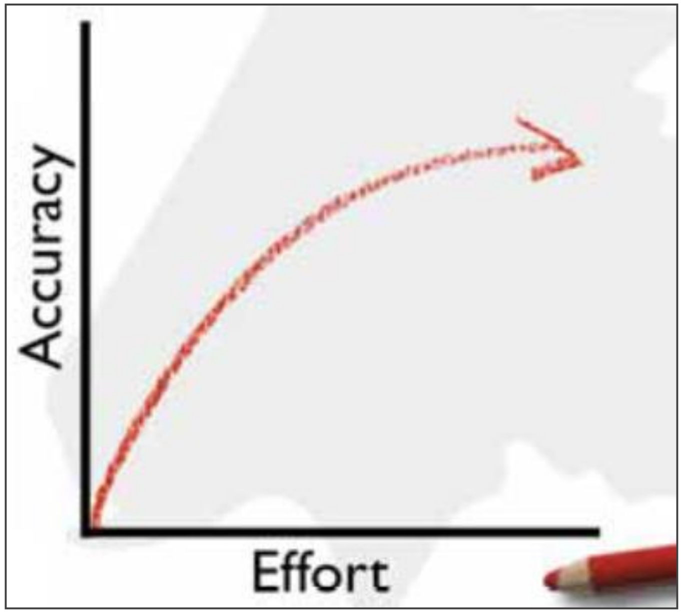
Top Challenges in Planning and Estimation

- Ambiguous / Changing Requirements
- Difficult to predict timelines of delivery
- Estimation is time consuming but still error prone
- Estimates are provided by the wrong people
- Software projects are unique and ambiguous, making accurate estimation difficult
- Business customers promise unrealistic deadlines and then expect the project team to make it work
- Commitments are signed based on high level estimates
- High level scope/design/concept and actual work holds huge difference
- Fixed scope, time and budget
- Estimates are expected to remain constant

Why Estimation Fails? (3/4)

Challenges in Estimation

- Estimation is one of the harder parts of a software Project
- Some data:
  - Nearly ⅔ of projects significantly overrun their cost estimates
  - 64% of the features included in products are rarely or never used
  - The average project exceeds its schedule by 100%
- Focused in features and not in activities



Why Estimation Fails? (4/4)

- Activities don't finish early
- Lateness is passed down the schedule
- Activities are not independent
- Features are not developed by priority
- We ignore Uncertainty
- Estimates become commitments

- Question?
  - Estimate for our project
    - One month for design and architecture
    - Four months for development
    - One month for testing
  - Scenario
    - Your first estimate is wrong by one week (design)
    - What do you do ?

Agenda

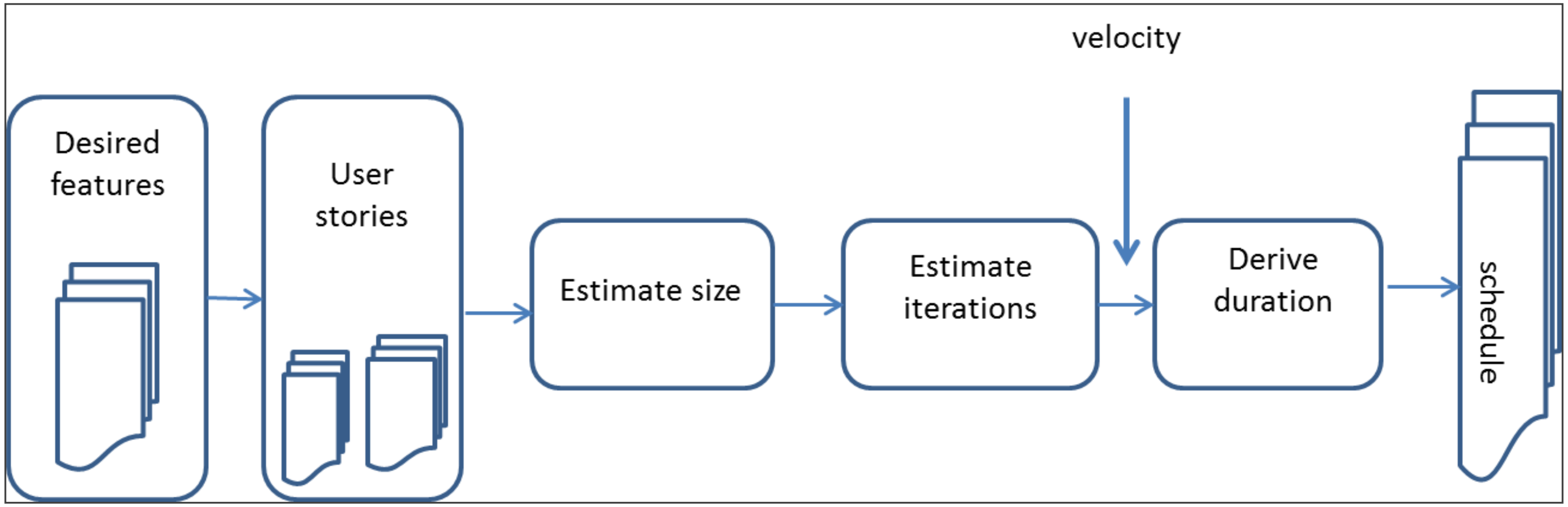
- Introduction
- **Estimation Process**
- Best Practices
- Antipatterns

Estimation Process

- How to Estimate?
- Estimation Techniques

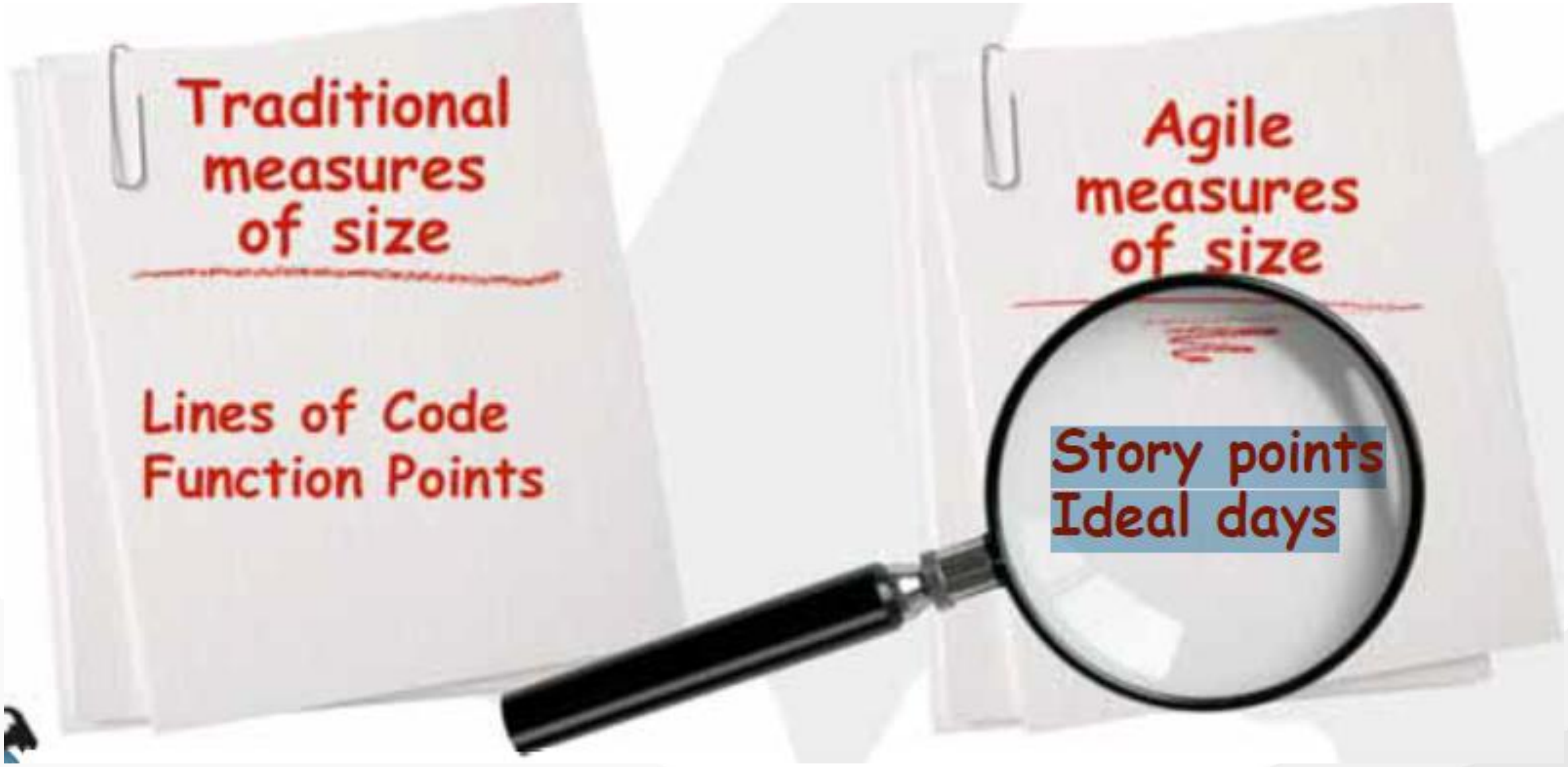
How to Estimate? (1/5)

Estimate size; derive duration



How to Estimate? (2/5)

Use Relative Measure of Size



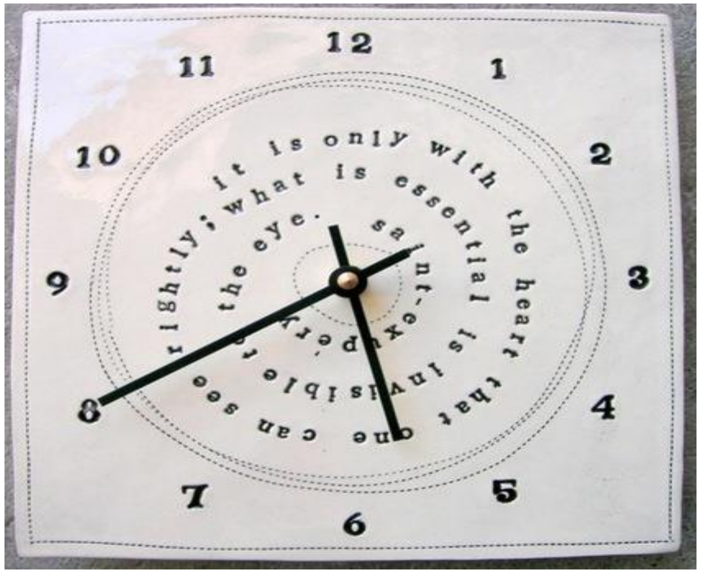
How to Estimate? (3/5)

- **Story Points**
  - The “bigness” of a task
  - Influenced by how hard/complex it is
  - Relative values are what is important:
    - A login screen is a 2
    - A search feature is an 8
  - Points are unit-less

- **T-Shirt sizing**
  - Based on complexity of the story
  - S, M, L , XL, XXL – Generic range

How to Estimate? (4/5)

- **Ideal Time**
  - How long something would take if
    - it’s all you worked on
    - you had no interruptions
    - and everything you need is available
  - 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

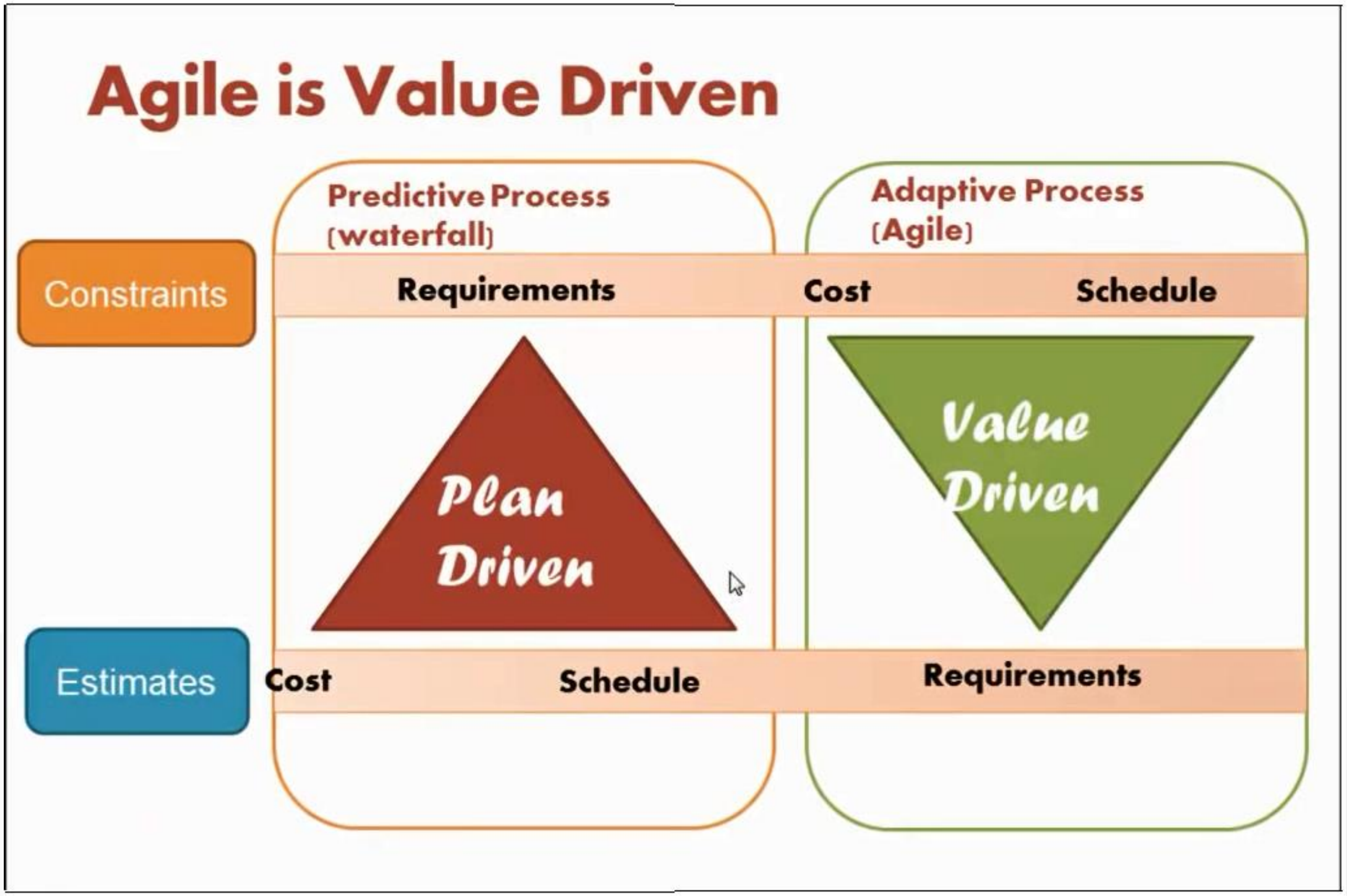

12
1
2
3
4
5
6
7
8
9
10
11
it is only with the heart that one can see rightly; what is essential is invisible to the eye.

How to Estimate? (5/5)

- **Comparing the Approaches**
  - Story points help drive cross-functional behavior
  - 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

Estimation Technique (1/6)

Predictive Process vs Agile Process



Estimation Technique (2/6)

- The three most common techniques for estimating are
  - Expert opinion
  - Analogy
  - Disaggregation
- Each of these techniques may be used on its own, but the techniques should be combined for best results

Estimation Technique (3/6)

Expert Opinion

- If you want to know how long something is likely to take, ask an expert
- **Advantages:**
  - It usually doesn't take very long
- **Disadvantages:**
  - On an agile project, estimates are assigned to user stories or other user valued functionality. Developing this functionality is likely to require a variety of skills normally performed by more than one person. This makes it difficult to find suitable experts who can assess the effort across all disciplines

Estimation Technique (4/6)

Analogy

- When estimating by analogy, the estimator compares the story being estimated with one or more other stories. If the story is twice the size, it is given an estimate twice as large
  - When estimating this way, you do not compare all stories against a single baseline or universal reference. Instead, you want to estimate each new story against an assortment of those that have already been estimated. To decide if a story should be estimated at five story points, see if it seems a little bigger than a story you estimated at three and a little smaller than a story you estimated at eight

Estimation Technique (5/6)

Disaggregation

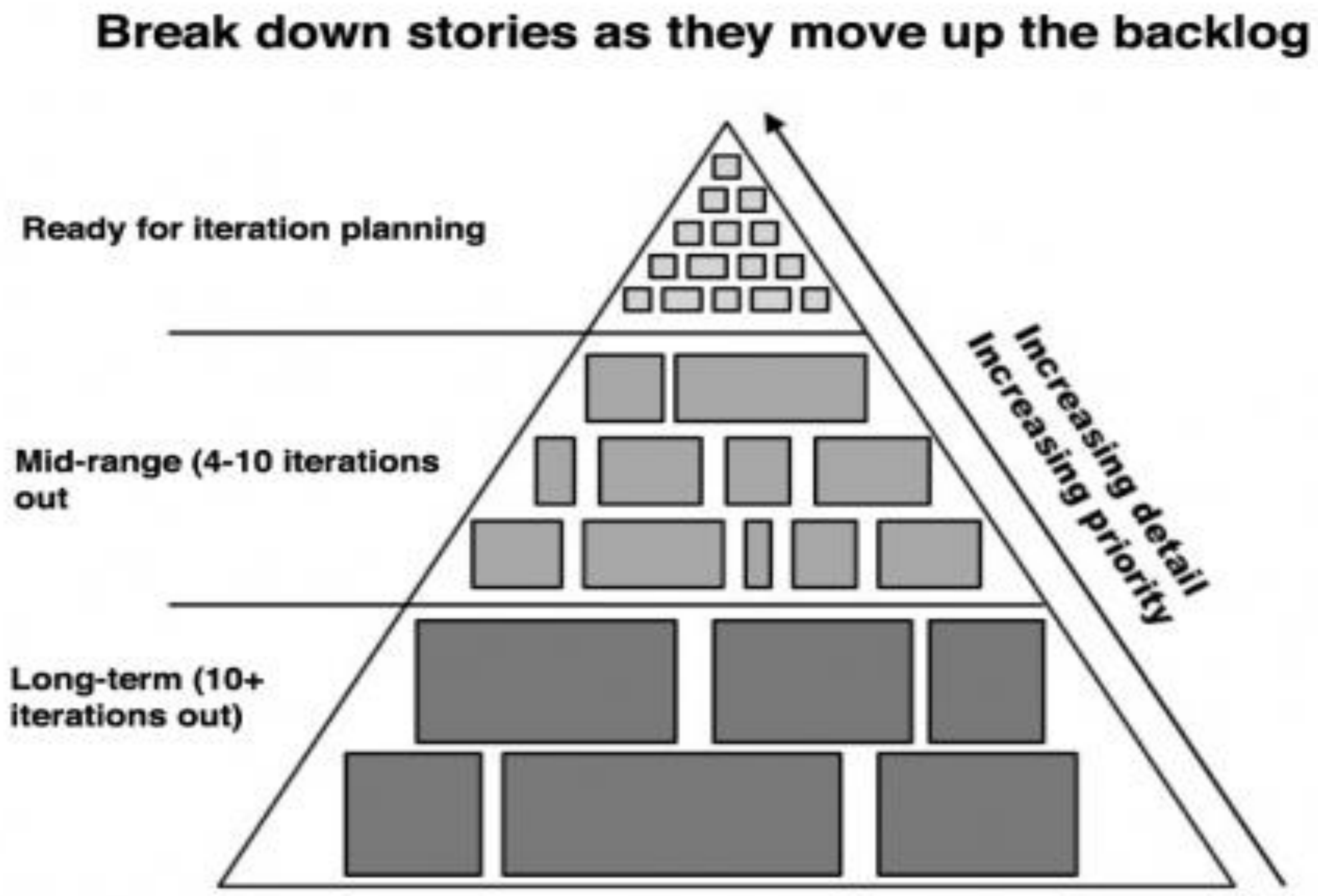
- Disaggregation refers to splitting a story or feature into smaller, easier-to-estimate pieces. However, you need to be careful not to go too far with this approach

Estimation Technique (6/6)

Factors Affecting Estimations

- The agility of the client organization
- Team maturity
- The mutual trust between the client and the offshore team
- Familiarity of domain and technical context
- Frequency of requirement change
- Unavoidable blockers
- Ad-hoc requests or unplanned activities getting priority

What if an Estimate is Large?

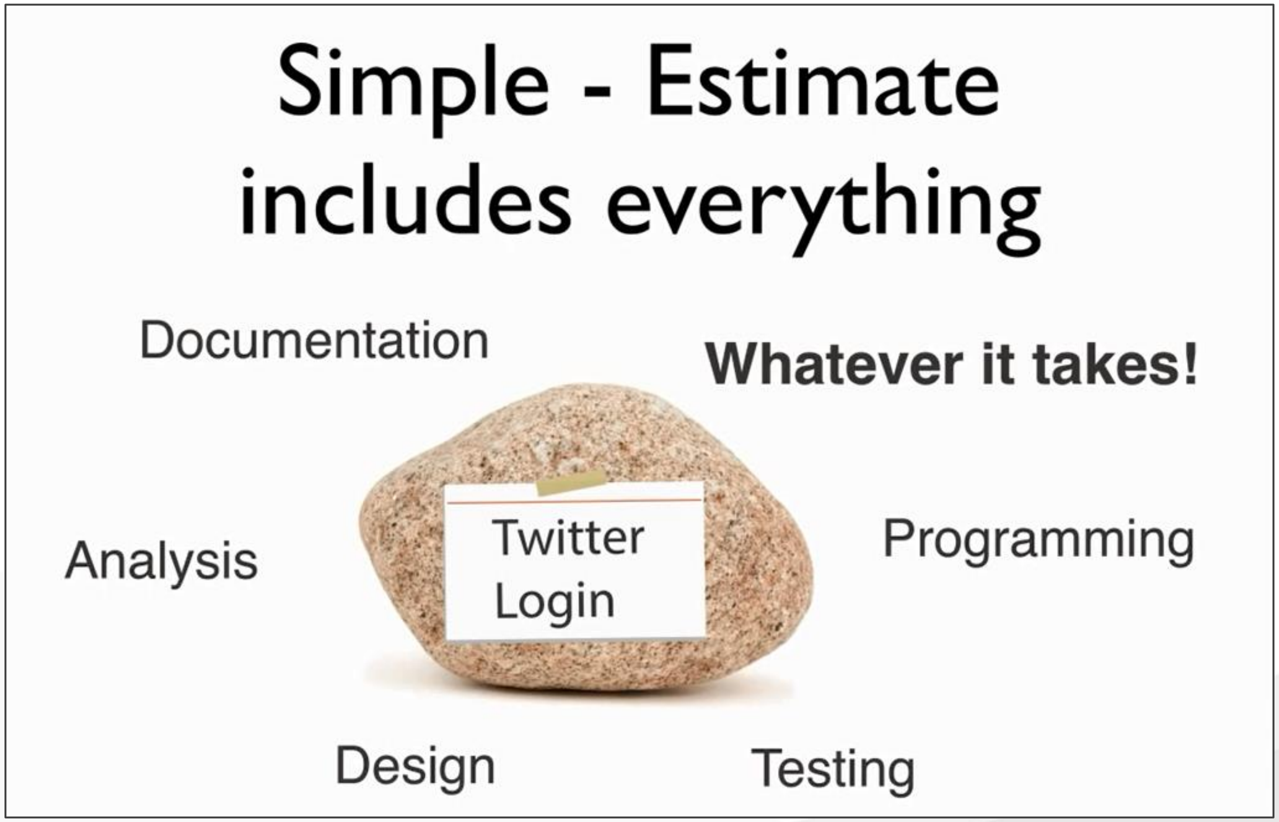

Break down stories as they move up the backlog
Ready for iteration planning
Mid-range (4-10 iterations out
Long-term (10+ iterations out)
Increasing detail
Increasing priority

Types of Estimation

- Absolute / Fixed / Simple
- Relative

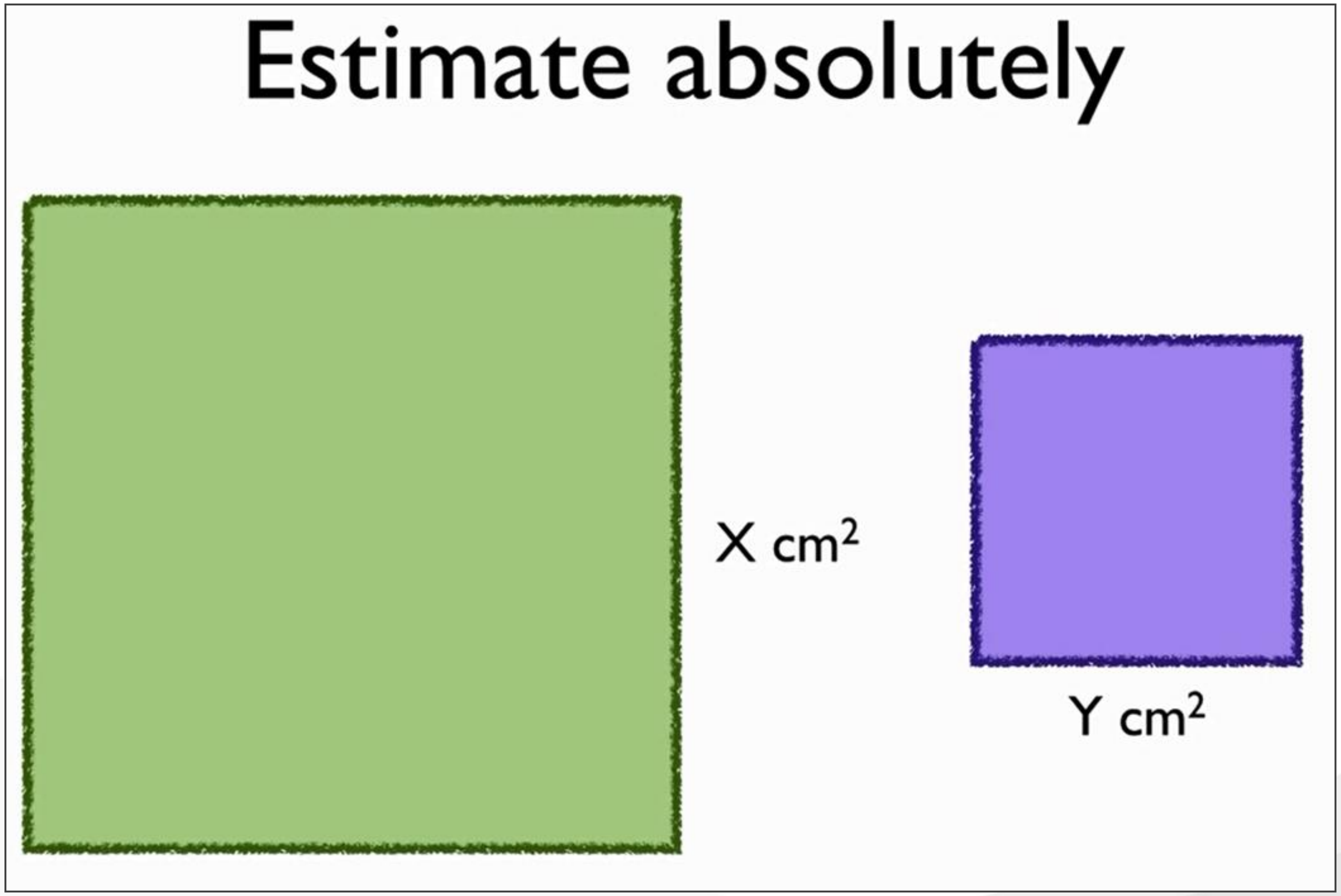
Absolute/ Fixed/ Simple (1/2)

- Assigning 1 number to the size of a story
- There are no separate estimates for analysis, design and testing
- It's one number that includes everything
- It is supposed to deliver everything necessary to make the story work



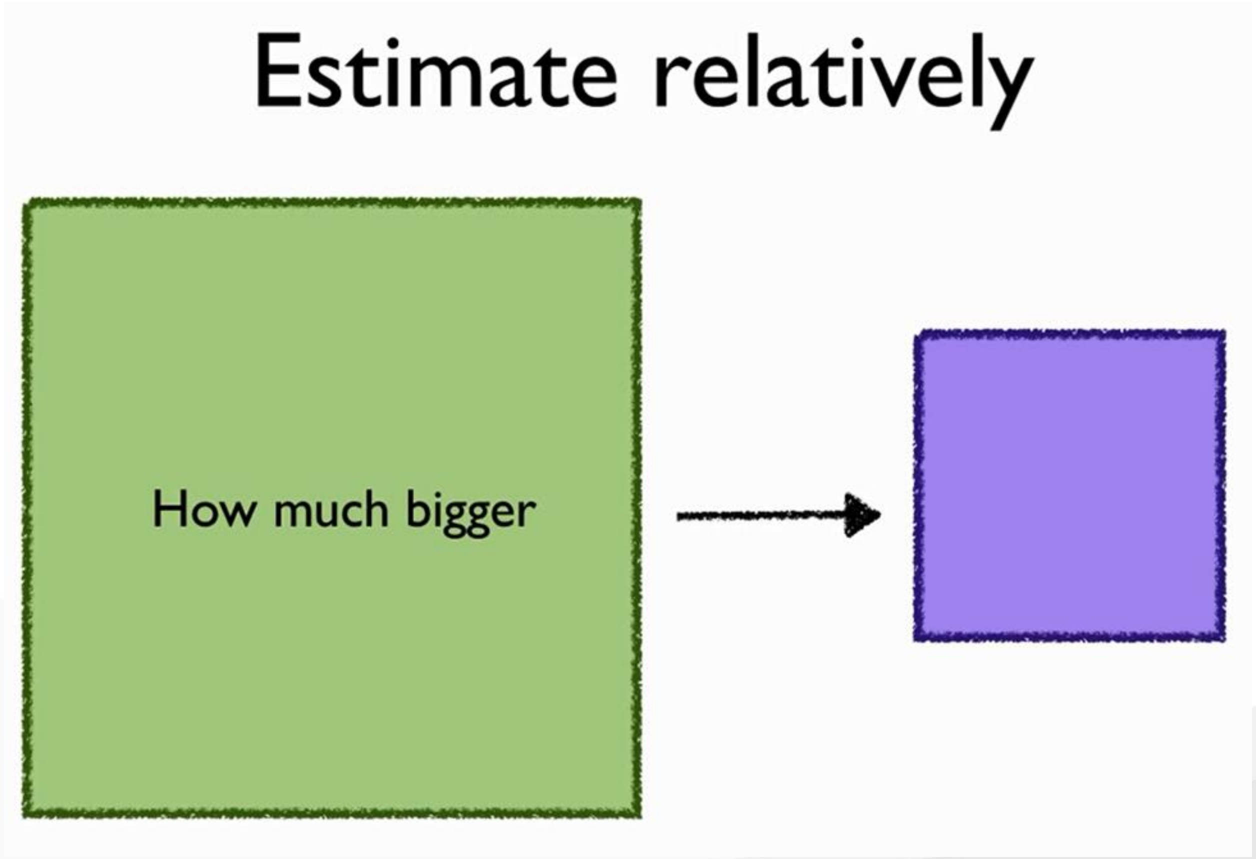
Absolute/ Fixed/ Simple (2/2)

- Estimating something as an exact round figure is tough
- It is difficult to exactly tell the size of both the squares below at a glance



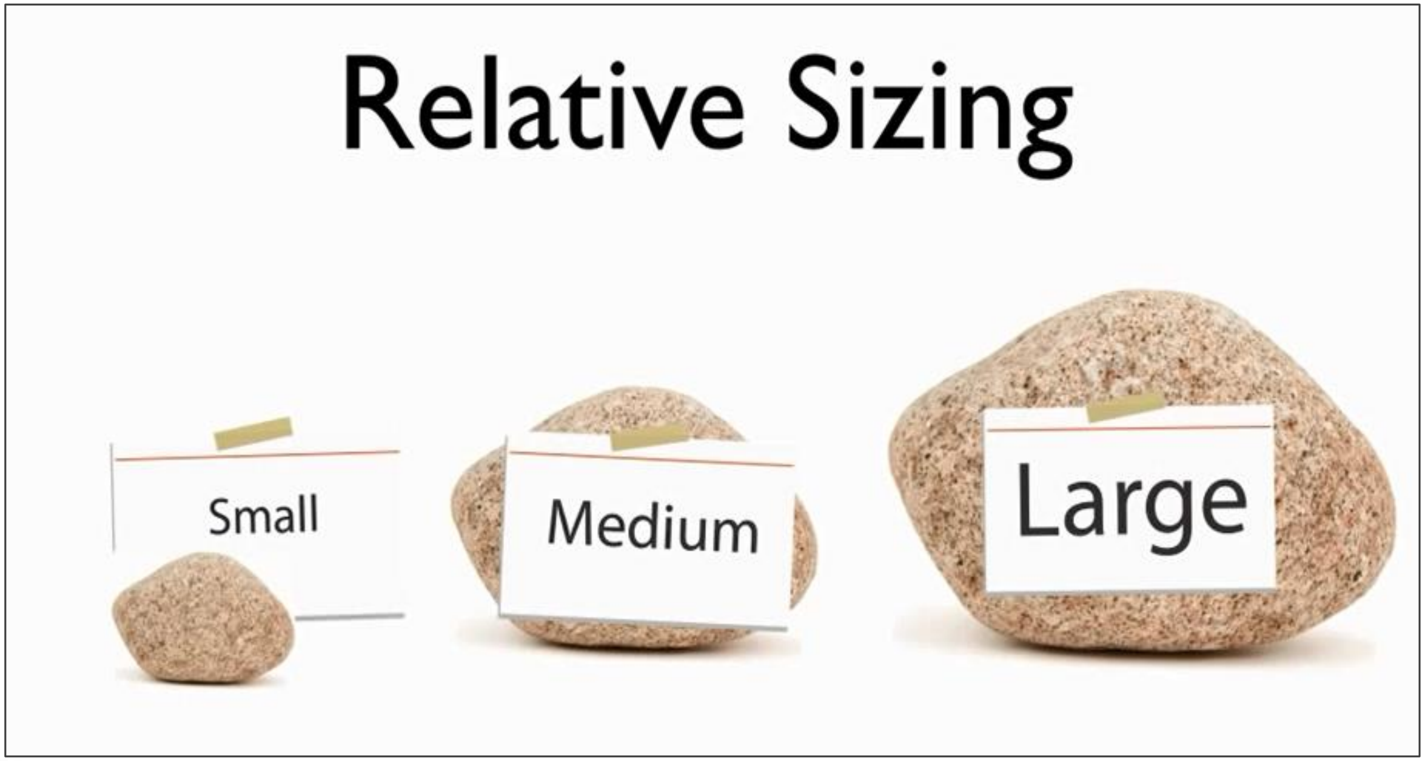
Relative (1/2)

- Estimating the size of one thing relative to another is easier
- It is easier to compare the 2 squares below and estimate that the purple square is about 1/4th of the green square
- Human beings can do relative estimation nicely



Relative (2/2)

- Comparing one story with another known story is an easier way to judge it’s size



Making it Abstract - Story Points

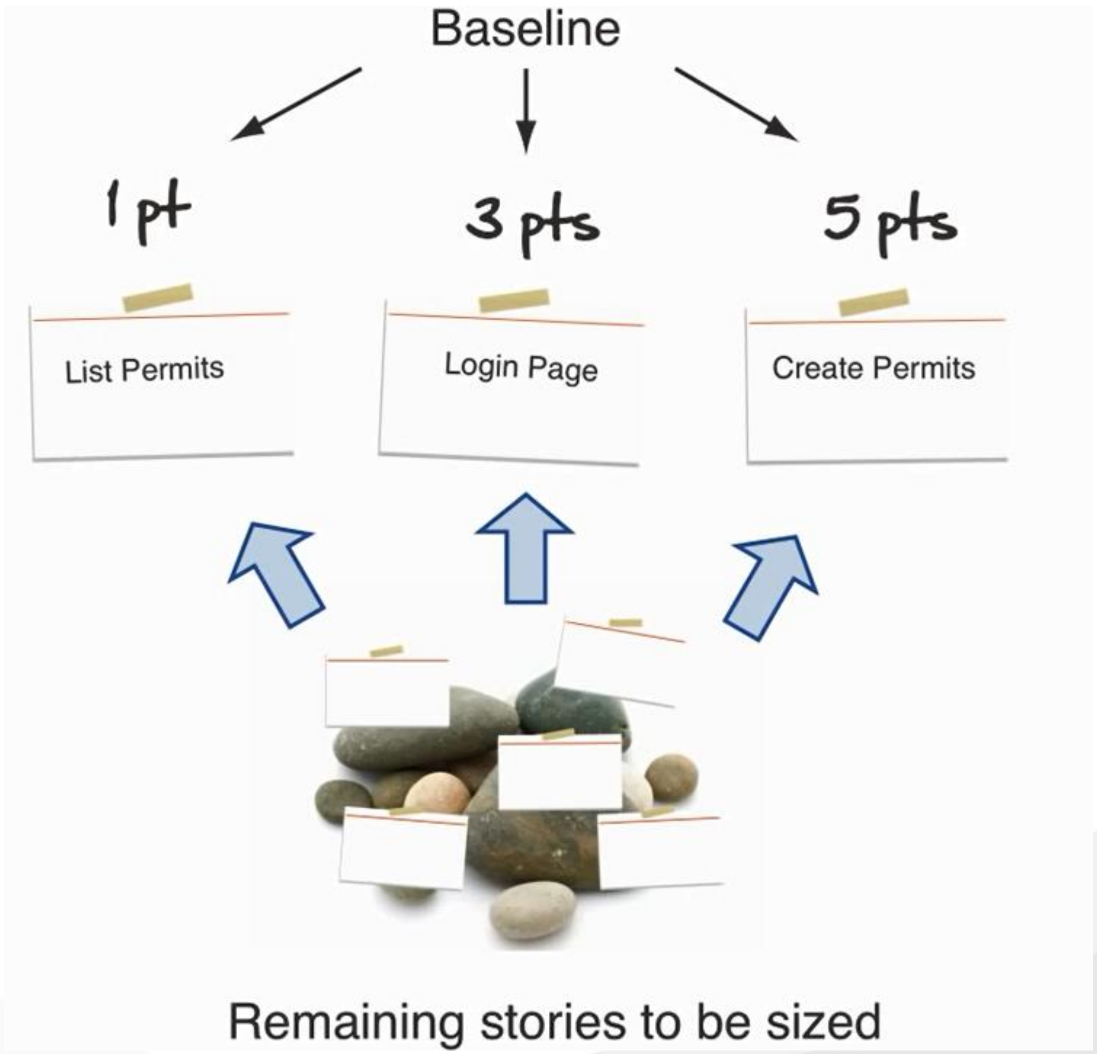
- Used to estimate the size of a story: the effort of a requirement
- Relative estimation - 2 story points require more effort than one story point and 13 story points require much more effort than 1 story point
- Attention to complexity: inherent and accidental

1 2 3 5 8 13 Sprint Planning

1 2 3 5 8 13 21 80 100 ...

Estimation using Story Points

- Instead of measuring tasks in days, the measurement should be done in terms of a points system based on relative complexity of each task



Planning Poker

- The best way I've found for agile teams to estimate is by playing planning poker (Grenning 2002)
- This method tries to make the meetings shorter and more productive, by making them more fun and dynamic

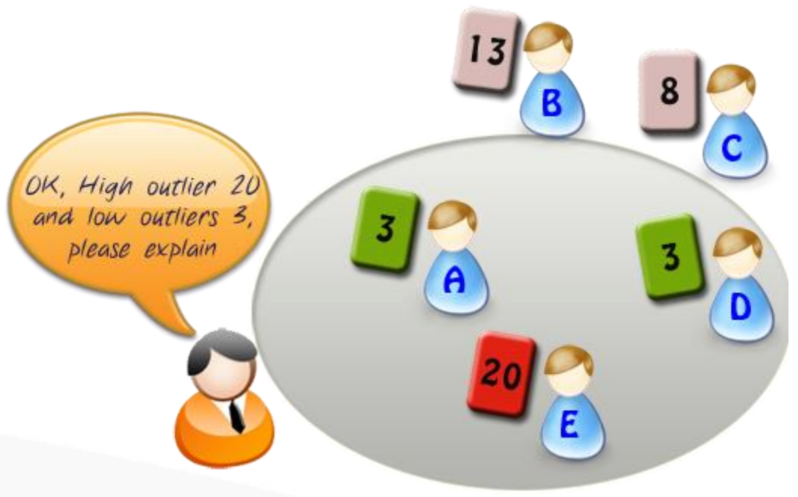
Planning Poker: The Meeting (1/3)

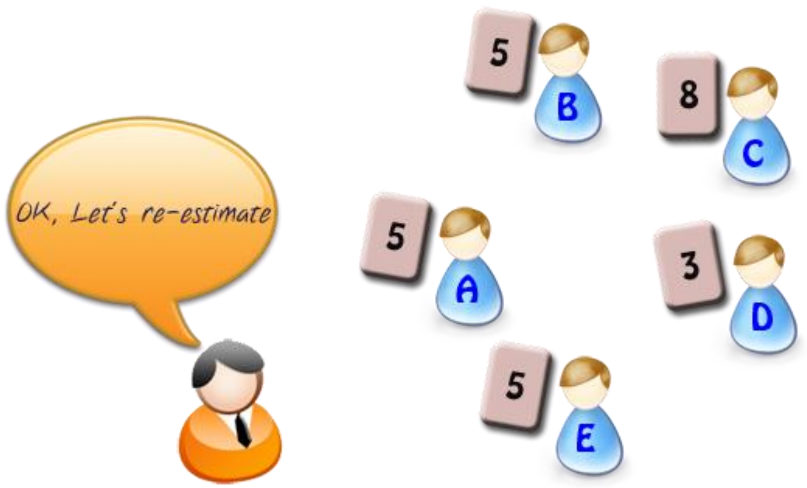
- A deck is given to each of the members
- The moderator exposes a user story in no more than 2 minutes
- Time for questions about the user story
- Each of the members choose a card privately (after giving thought on story understanding and required efforts)
- Once everybody has chosen, all the cards are turned over at the same time
- In this first round, it's probably that the estimations will differ significantly

What is the size of this user story?
13
B
8
C
3
A
3
D
20
E

Planning Poker: The Meeting (2/3)

- In case the estimations differ, the high and low estimators expose their reasons
- A few minutes for the team to discuss about the story and the estimation
- Again, each member thinks privately a estimation, and they show the cards simultaneously
- If the estimations still differ, the same process can be repeated





Planning Poker: The Meeting (3/3)

- When the estimations converge, the process finishes, and the next user story is estimated
- In case the estimations don't converge by the 3rd round, there are some options:
  - Leave the story for now and try again later
  - Ask the user to decompose the story in smaller parts
  - Take the highest, lowest or average estimation

Velocity

- The rate at which a team can produce working software
- Measured in non-time-referent terms (e.g., Story Points) per Sprint
- More accurately stated, it is measured in terms of the stabilized number of Story Points a team can deliver per sprint of a given length, and with a given definition of Done.
- Used for estimation and planning
- Must have stabilized to be reliable
- Should not be used as a measure of comparison across teams
- **An Example**
  - Team A is working in 2-week sprints on work that it has estimated together. Team A has been working together for several sprints, and consistently delivers between 18 and 23 points of working software every sprint.
  - We could reasonably expect Team A to deliver roughly 20 points per 2-week sprint, and so we consider that to be the team's velocity for planning purposes.
  - If there are eight 2-week sprints in a release, we can extrapolate that Team A has the capability to deliver 160 points in a release.

Agenda

- Introduction
- Estimation Process
- **Best Practices**
- Antipatterns

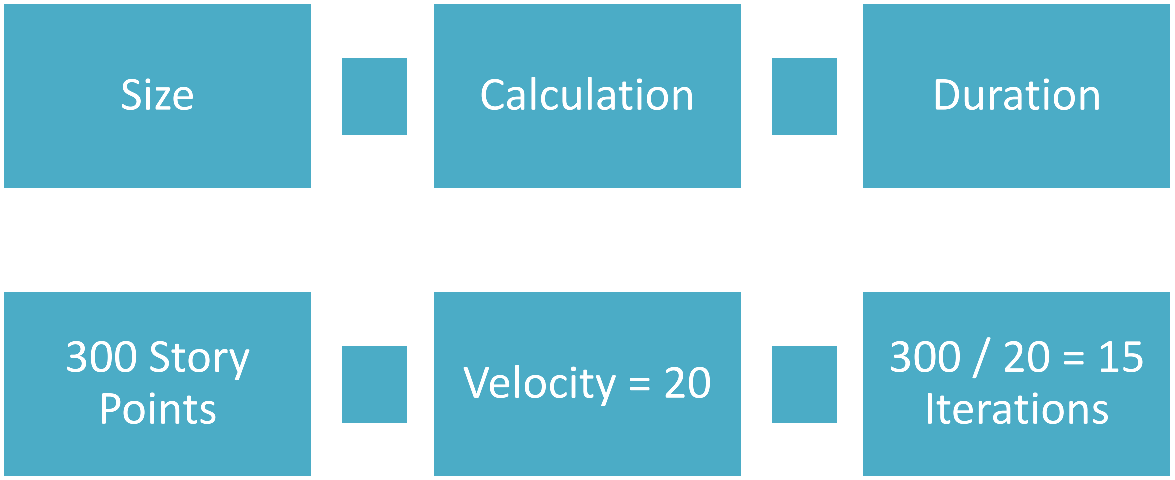
Best Practices

- Use more than one person
- Use more than one approach
- Know when to stop
- Do not over estimate
- Present estimates as a range
- Defend / explain estimate range probabilities
- Don’t reserve estimating for when you know least about the project
- Be aware of common estimation omissions
- Embrace reality early
- Review, Revisit, Repeat

I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices
I will adopt Best Practices

Summary

- Don’t forget an estimate is only an estimate.
  - The better you are at it, the better you will be at hitting deadlines and protecting quality
- Estimate size; derive duration



Estimation Ball Game

Rules

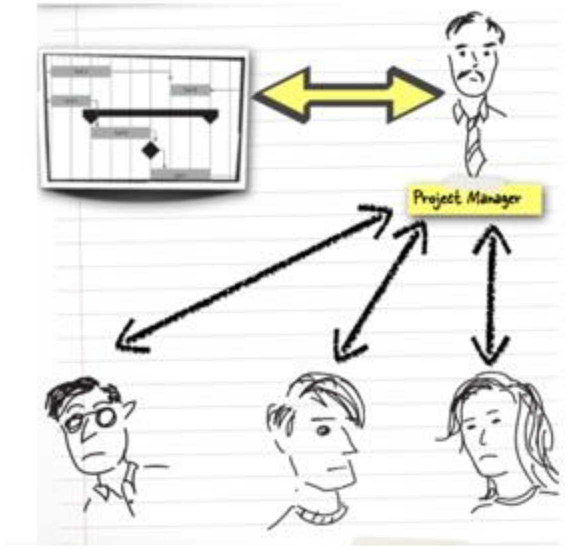
- Each ball must have air-time.
- Each ball must be touched at least once by every team member.
- Balls cannot be passed to your direct neighbor to your immediate left or right.
- Each ball must return to the same person who introduced it into the system.

Agenda

- Introduction
- Estimation Process
- Best Practices
- **Antipatterns**

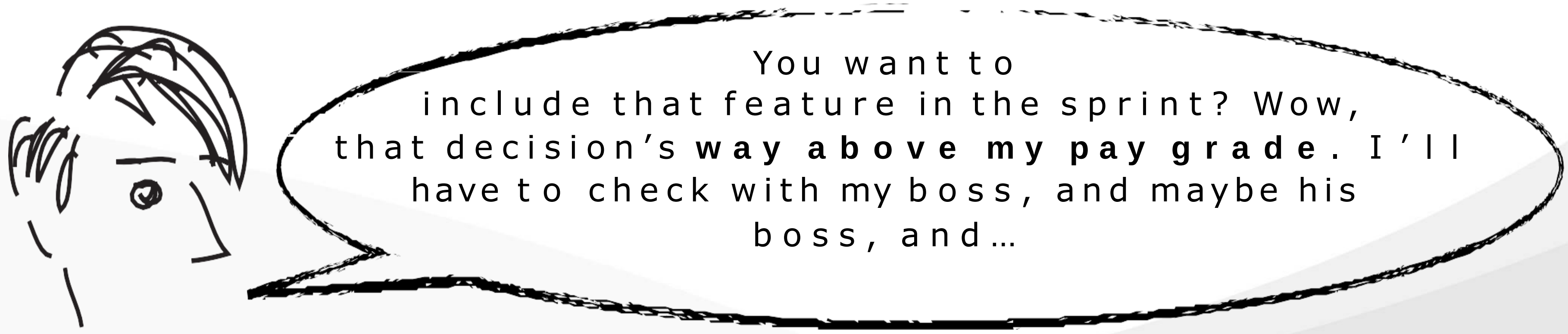
Antipattern: Command-and-control Scrum

- Command-and-Control project managers feel like they create and "own" the schedule by themselves
- Someone in authority estimates the stories instead of team
- Team Leads assign work to the team members to do exactly what's required
- This can lead to "water-Scrum-fall": the project is planned up front, and iterations are just project phases
- Teams don't commit to Sprint Goal
- Team members added/removed to scrum teams at random and frequently



Antipattern: Product Owner without authority

- The Product Owner decides what items go into the product backlog and which of those items the team can work on during each sprint. At the end of the sprint, the Product Owner accepts the completed items on behalf of the company. This depends on the Product Owner having the authority to make those decisions. That authority is compromised if:
  - The team elects a Product Owner from within their own ranks who does not have any special authority
  - Senior management chooses a Product Owner but doesn't stand behind POs decisions
  - The Product Owner role is replaced with a committee

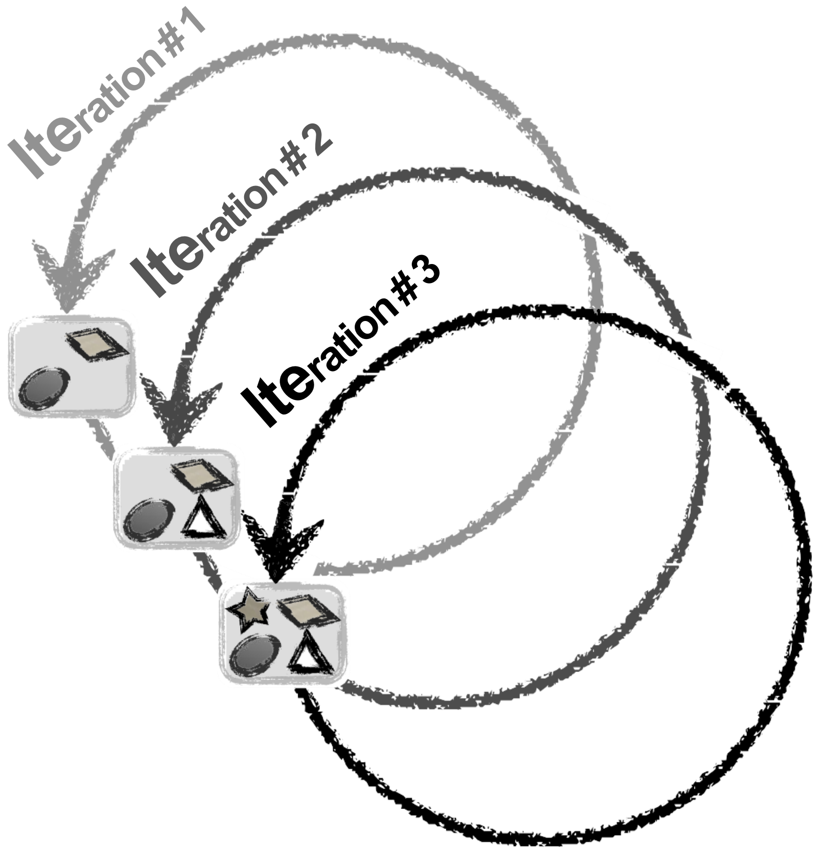

You want to include that feature in the sprint? Wow, that decision's **way above my pay grade.** I'll have to check with my boss, and maybe his boss, and…

Antipattern: Sprints aren't really iterations

Scrum is an iterative methodology because it uses iterations called sprints. The team demonstrates working software to the users at the end of each iteration. Iterations are timeboxed, which means the deadline does not change. Work that isn't completely "Done" done at the end of the sprint is put back in the product backlog and resumed in a later iteration.

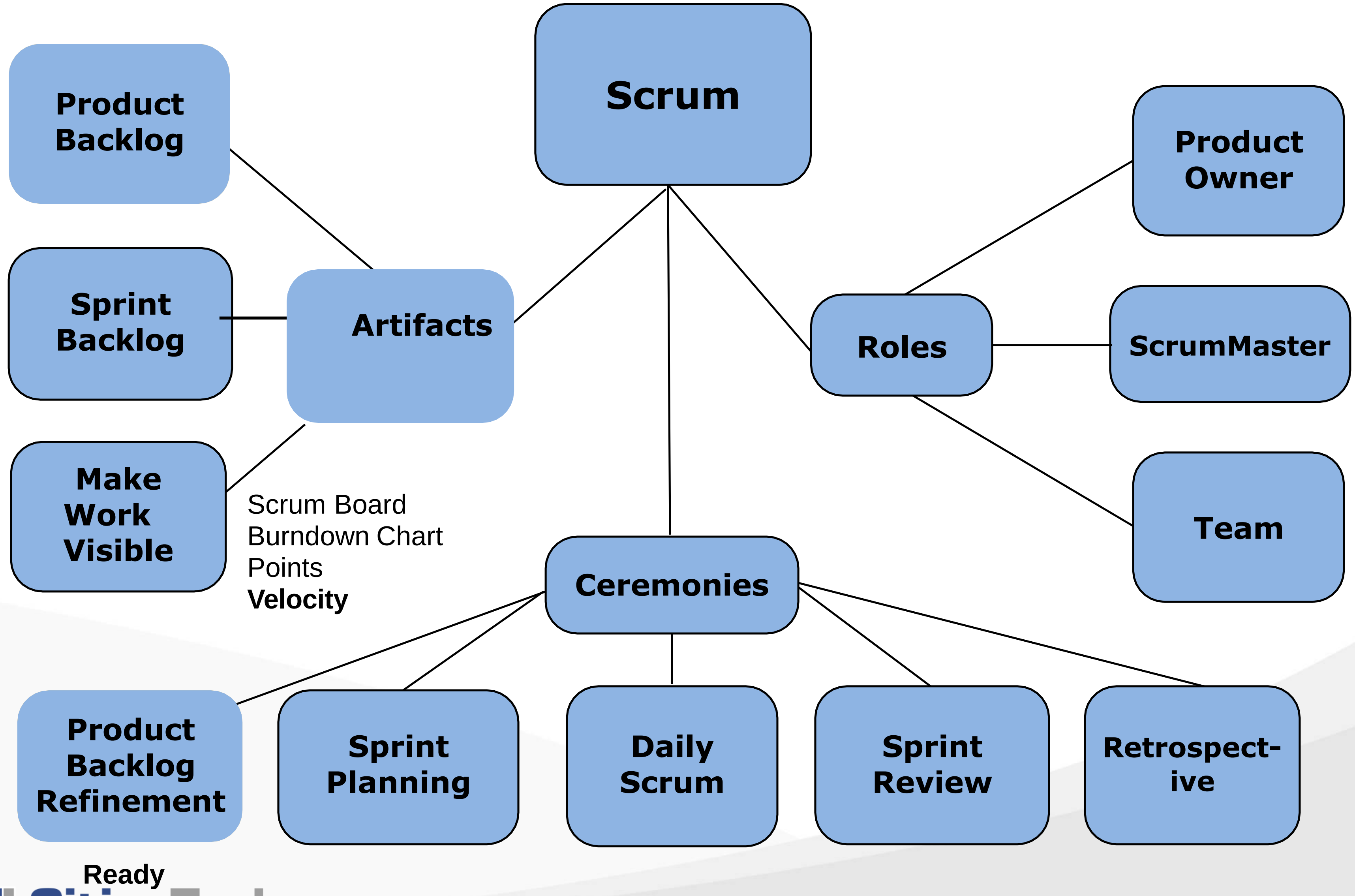
But a lot of Scrum teams use sprints that aren't really iterations:

- The deadline moves to accommodate unfinished work
- The team will demo work that isn't really "Done" done
- Features move in and out of the sprint seemingly at random
- People aren't working together towards a shared sprint goal



CitiusTech

Connecting the dots



Thank You

CitiusTech Markets

CitiusTech Services

CitiusTech Platforms

Accelerating Innovation

CitiusTech Contacts

Email ct-univerct@citiustech.com

www.citiustech.com

CitiusTech