Agile Toolbox: The Product Backlog

Class will begin at 11am Eastern Time

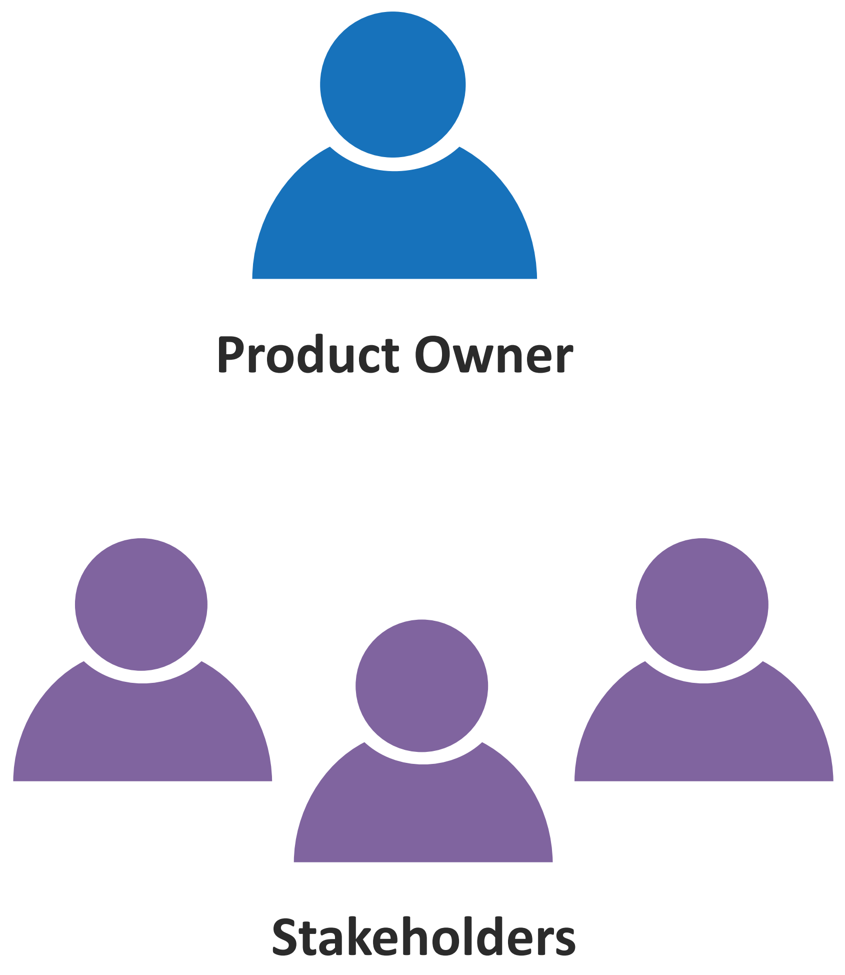
Product Backlog

The Product Backlog is an ordered list of what is needed to create or improve a product. It is the single source of work undertaken by an agile team.

PRODUCT OWNER

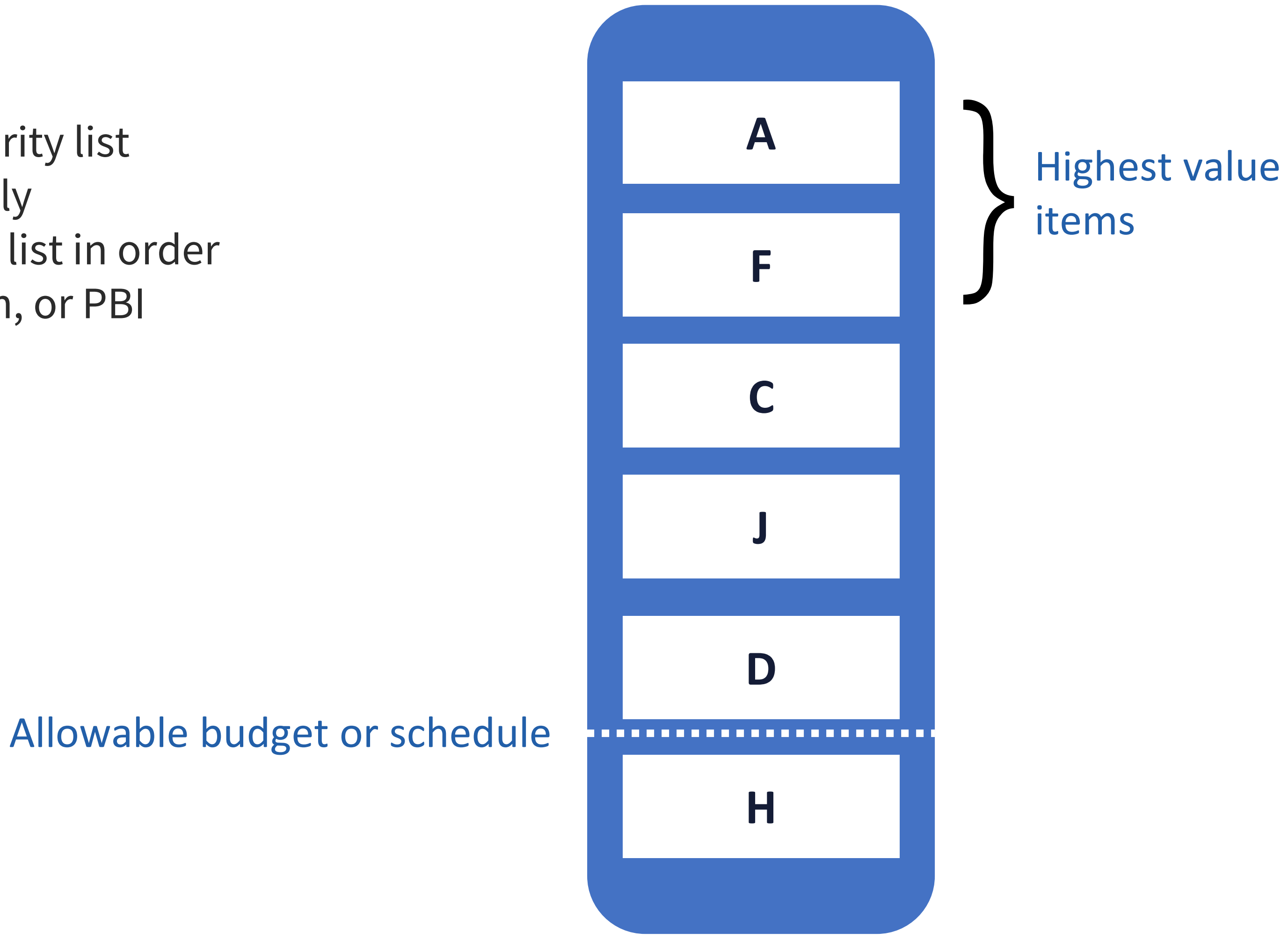
- Manages and prioritizes the product backlog
- Collects requirements from stakeholders
- Serves as voice of the stakeholders (liaison)
- Develops product vision and ensures transparency
- Communicates the product goal
- Ensures value delivery
- Controls the budget

Note: The title is not important. Anyone can manage a backlog, and a backlog is not solely associated with new product development.



Product Backlog

- Product Owner decides priority
- Highest value items on top
- New features can be inserted into the priority list
- Represents the scope, shown incrementally
- Negotiations and trade-offs help keep the list in order
- Each item is called a Product Backlog Item, or PBI
- A common PBI is a **"User Story"**
- All work should be included
  - New features and functionality
  - Maintenance work
  - Bug fixes
  - Changes
  - Single, prioritized list



User Stories

Short, simple descriptions of a feature

Told from the user's perspective

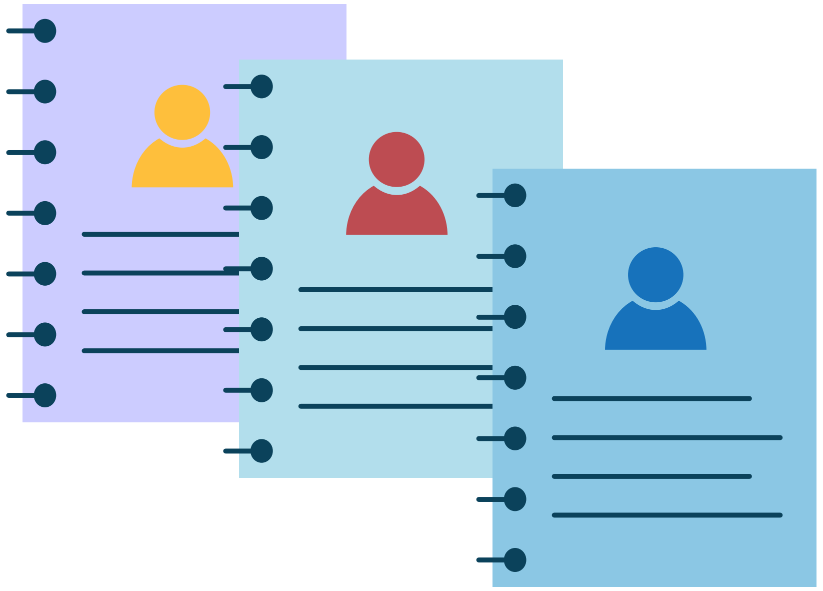
When large or complex, can be called "epics"

Sentence structure:

"As a *role*, I want *functionality*, so that *business benefit*."

Example:

"As a customer, I want my credit card information to be stored, so that I save time when checking out."



INVEST Criteria for Effective User Stories

Independent - developed in any order

Negotiable - discussions with Product Owner

Valuable - justify the work

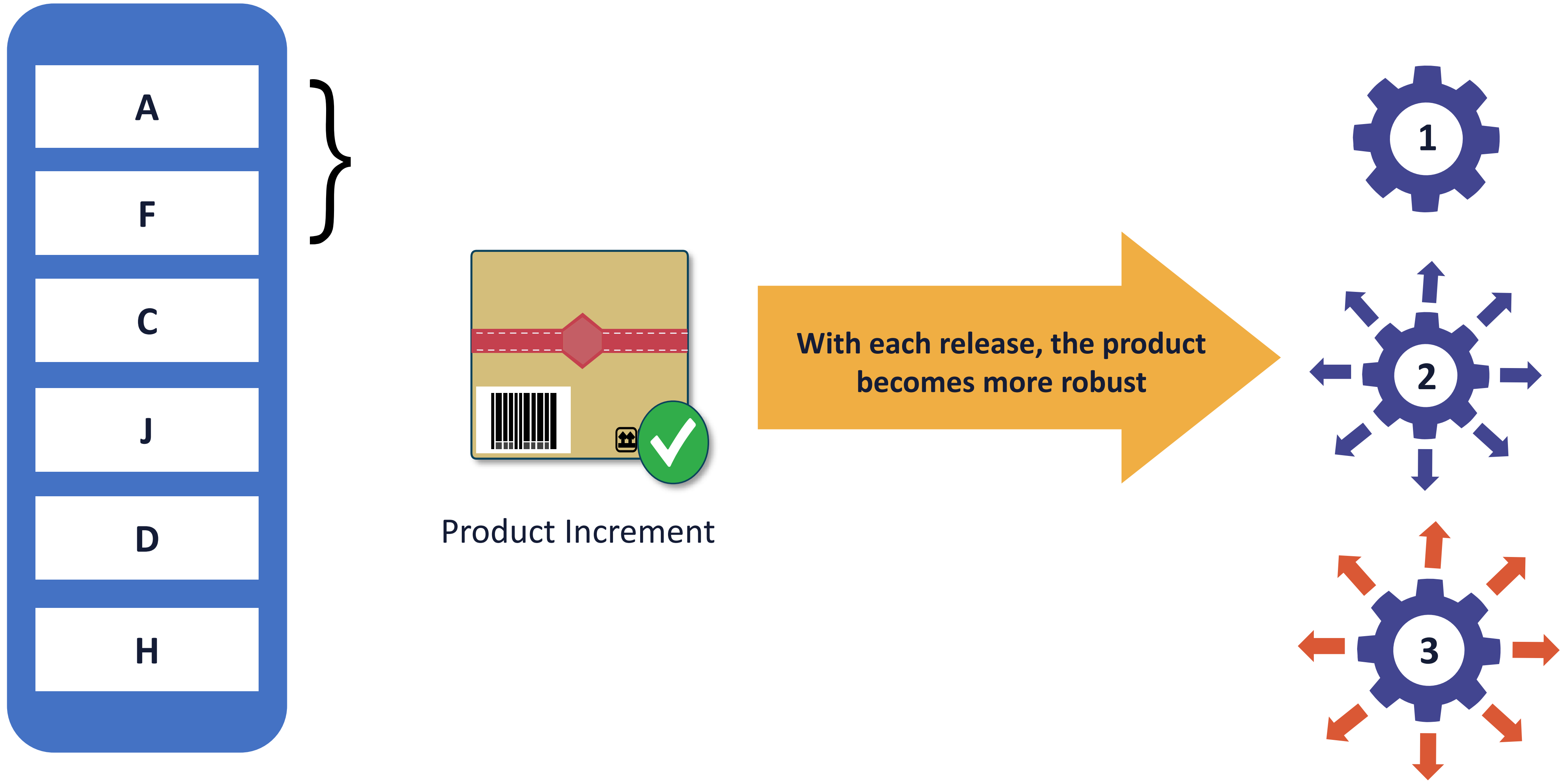
Estimatable - quantify the effort

Small – reliable estimates of 4-40 hours of work

Testable – measure progress and acceptance

I N V E S T

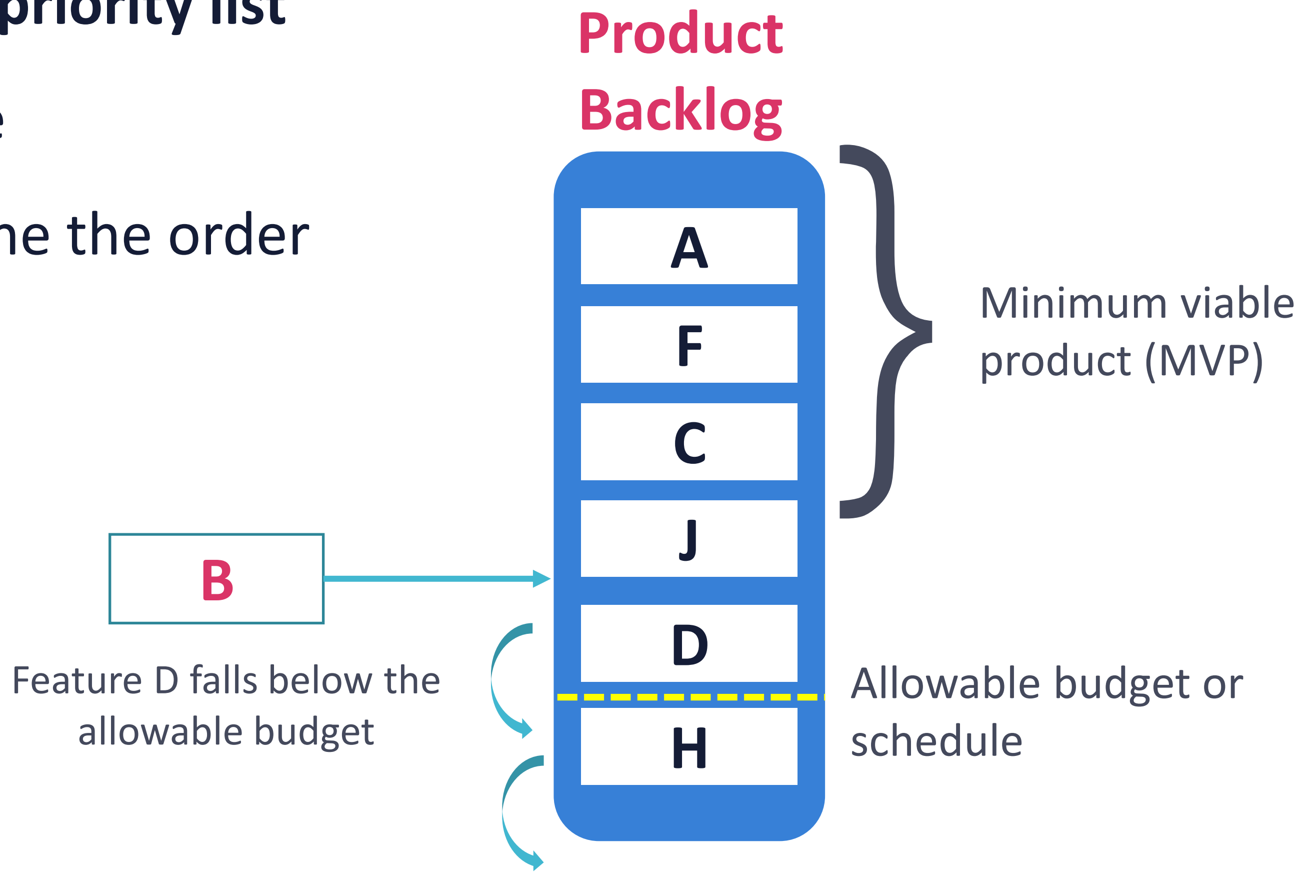
Incremental Delivery
Product Scope Grows and Evolves



Scope Changes in Agile Projects

New features can be inserted into the priority list

- The budget does not need to change
- Trade-offs and negotiations determine the order

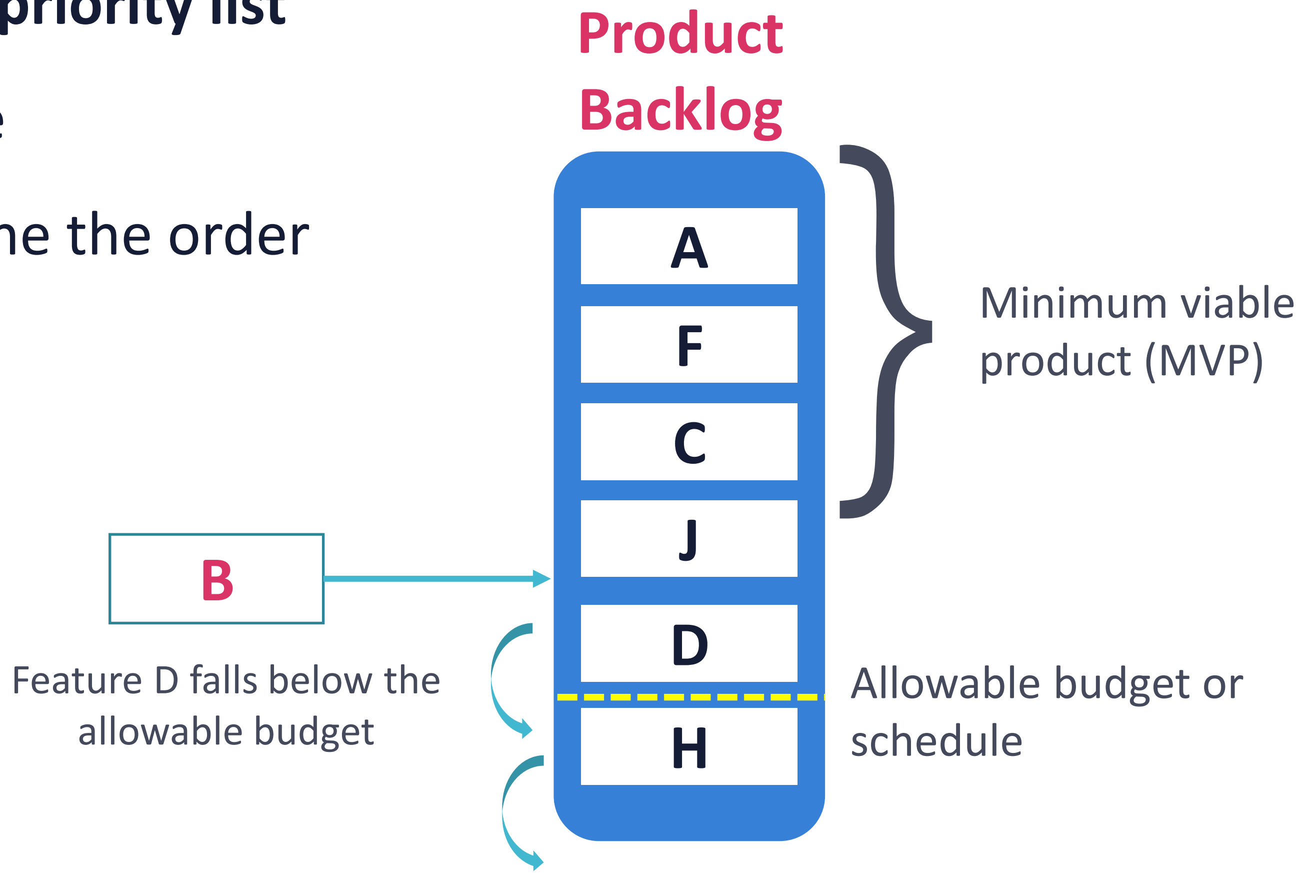

Product Backlog
A
F
C
J
D
H
B
Minimum viable product (MVP)
Allowable budget or schedule
Feature D falls below the allowable budget

Scope Changes in Agile Projects

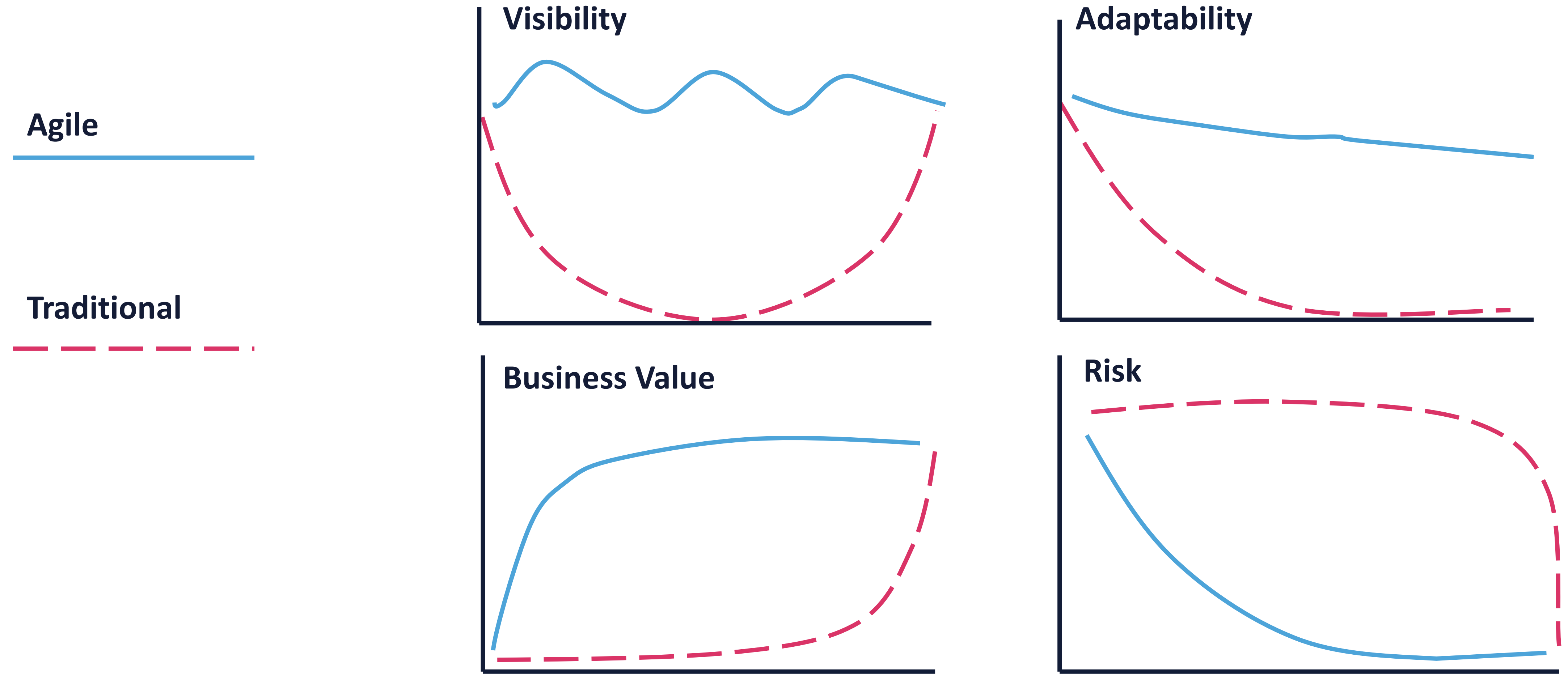
New features can be inserted into the priority list

- The budget does not need to change
- Trade-offs and negotiations determine the order

Has your list changed today, based on new priorities or new information?


Product Backlog
A
F
C
J
D
H
B
Minimum viable product (MVP)
Feature D falls below the allowable budget
Allowable budget or schedule

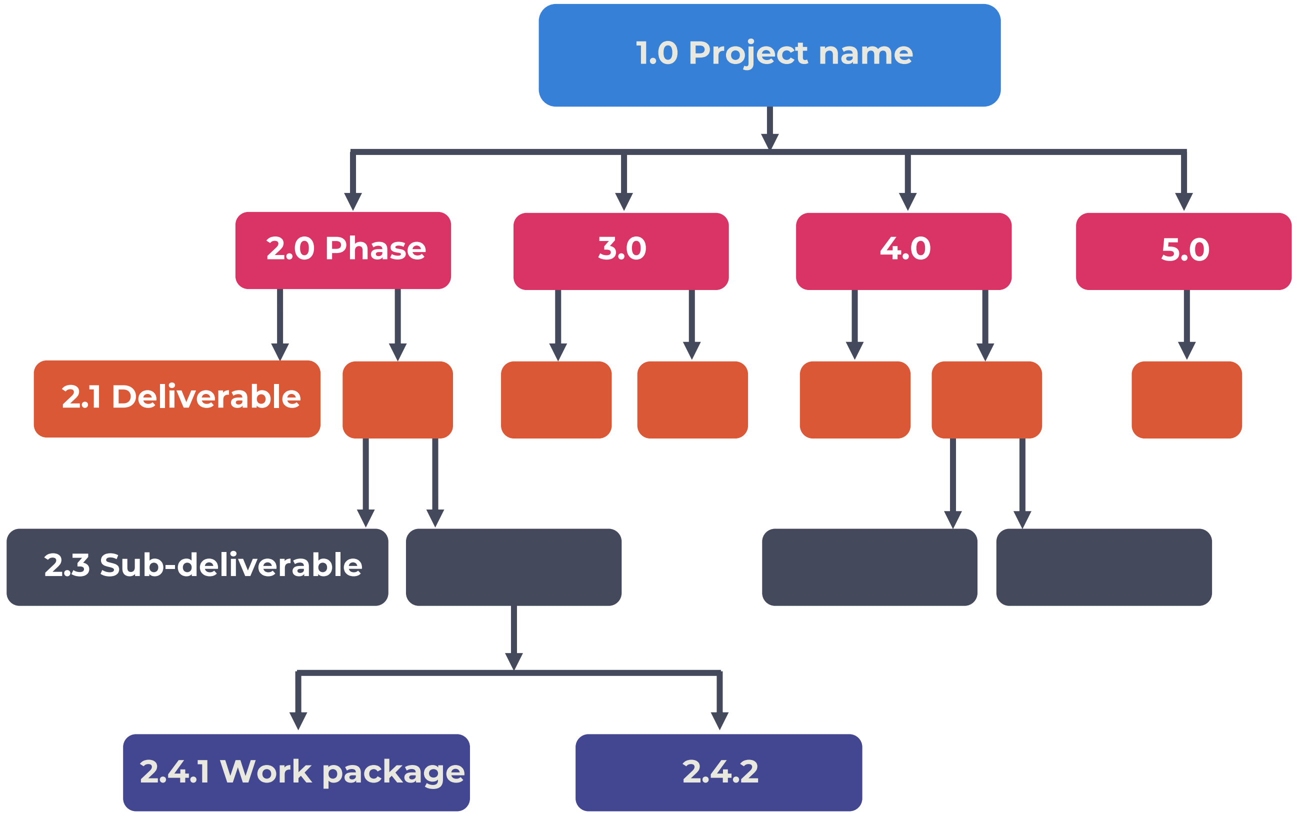
The Agile Value Proposition



Comparison of Agile and Traditional Projects

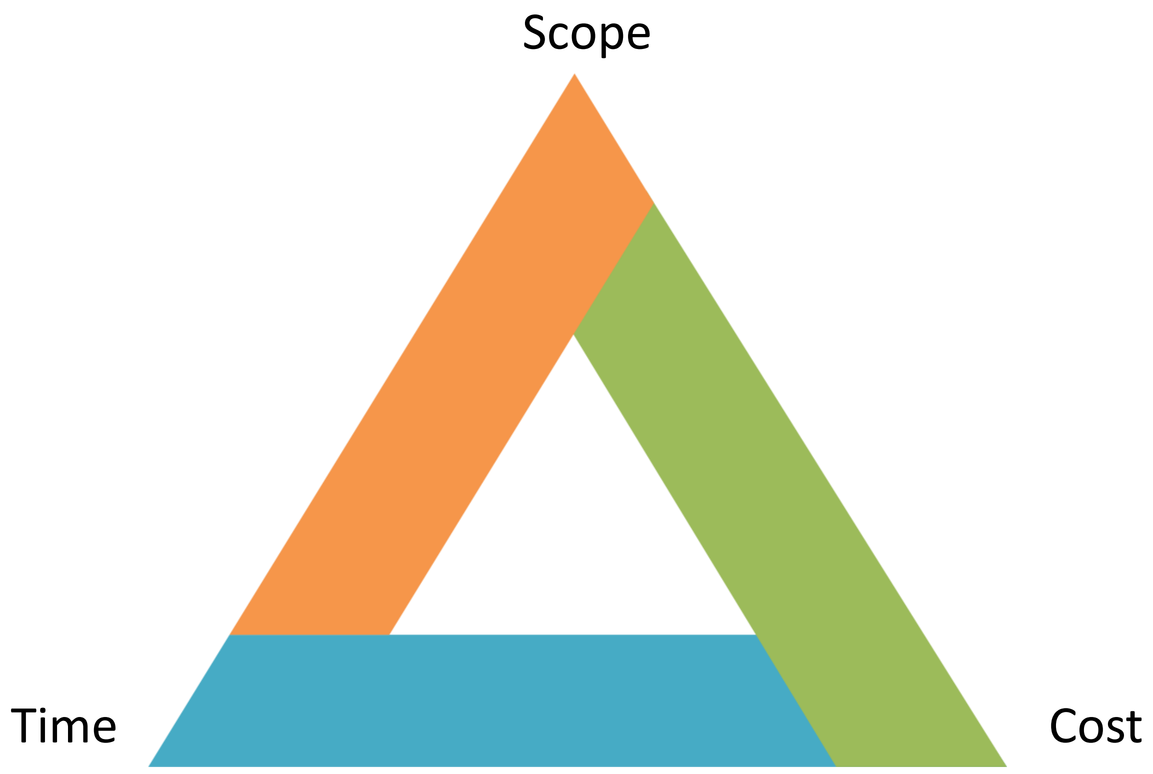
Work Breakdown Structure (WBS)

A Tool for Decomposing Work in Traditional Projects

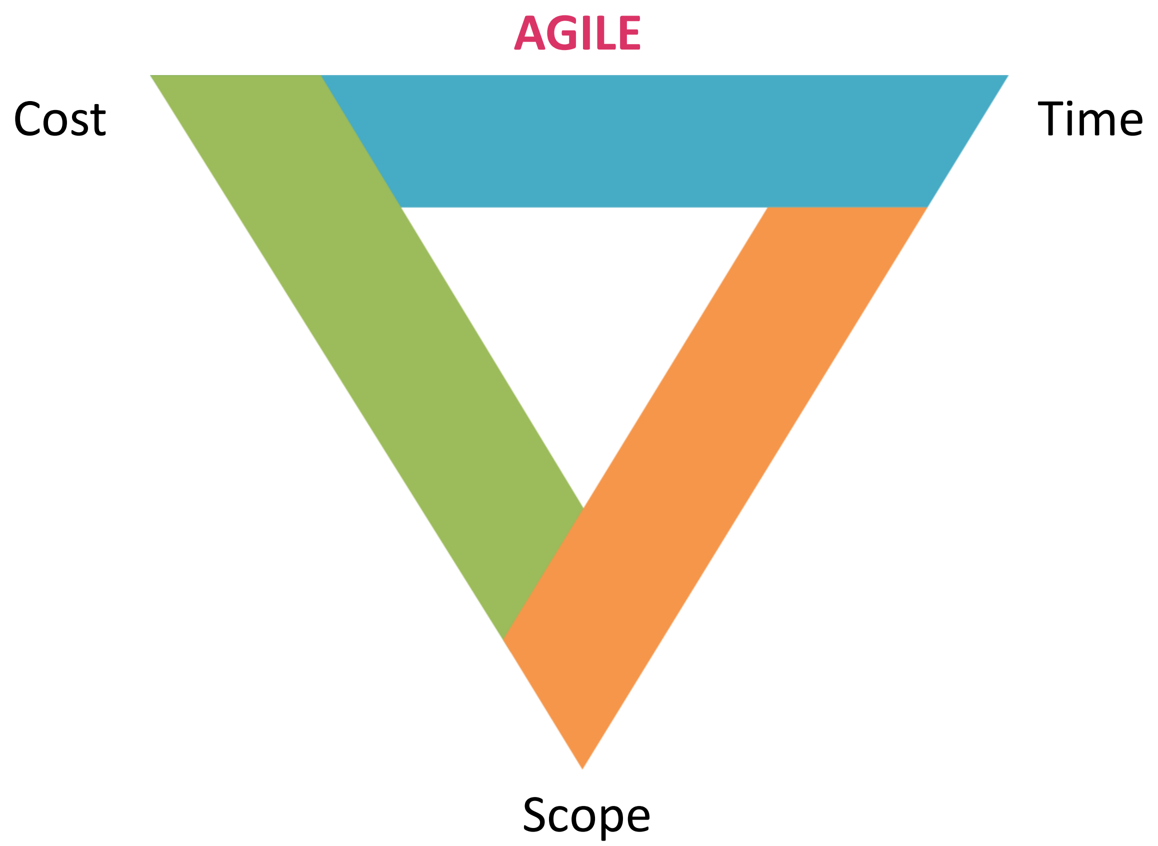


- Decomposition
- Lowest level is always called the "work package"
- Complete project scope is known

The Triple Constraints



The Agile Inverted Triangle



Product Backlog Exercise

Review your personal to-do list

- Tell us a few items on your list
- Are the most important items on top?
- What other factors determine the order of your list?

Product Backlog Exercise

Review your personal to-do list

- Tell us a few items on your list
- Are the most important items on top?
- What other factors determine the order of your list?

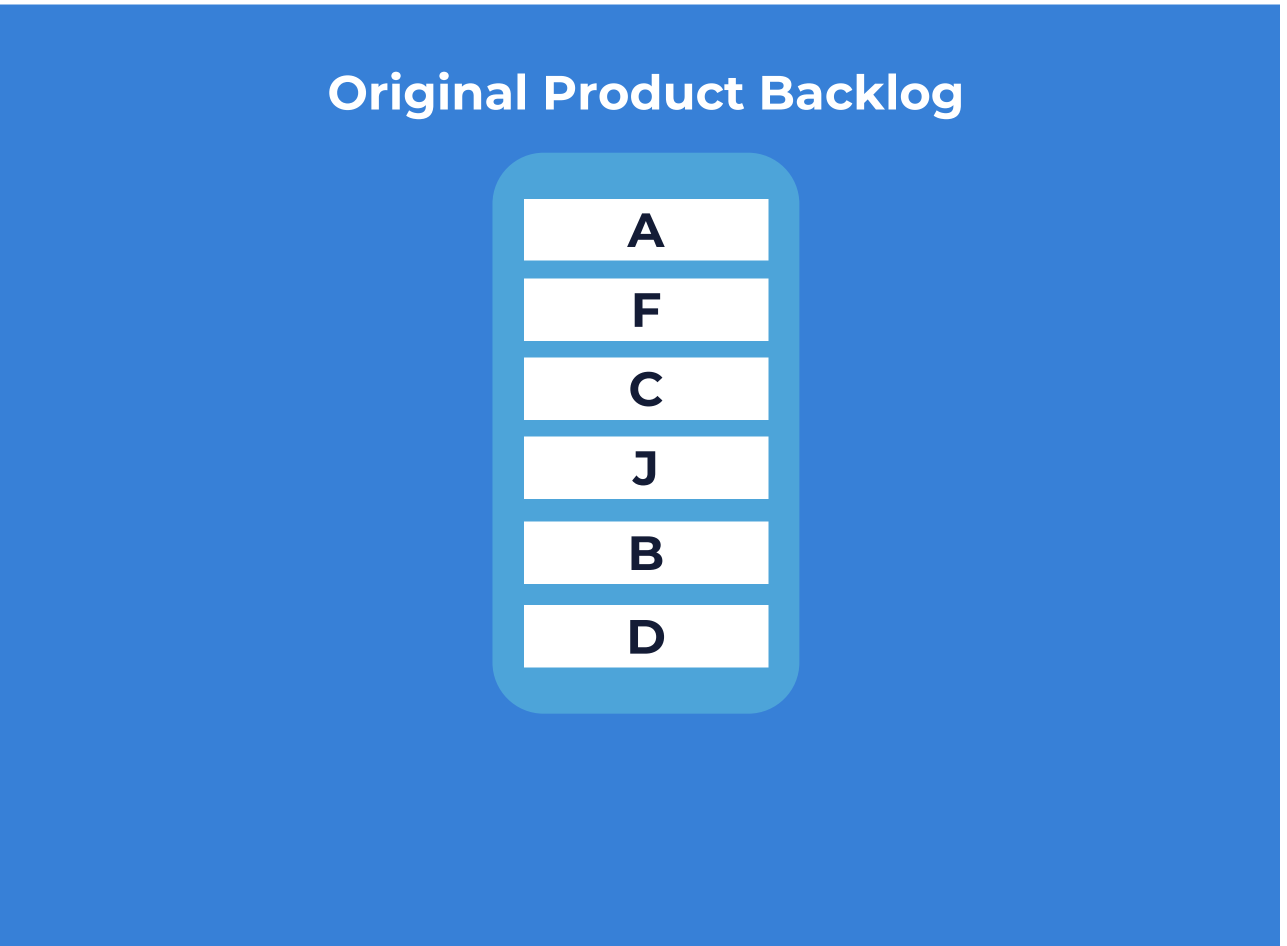
New constraint!

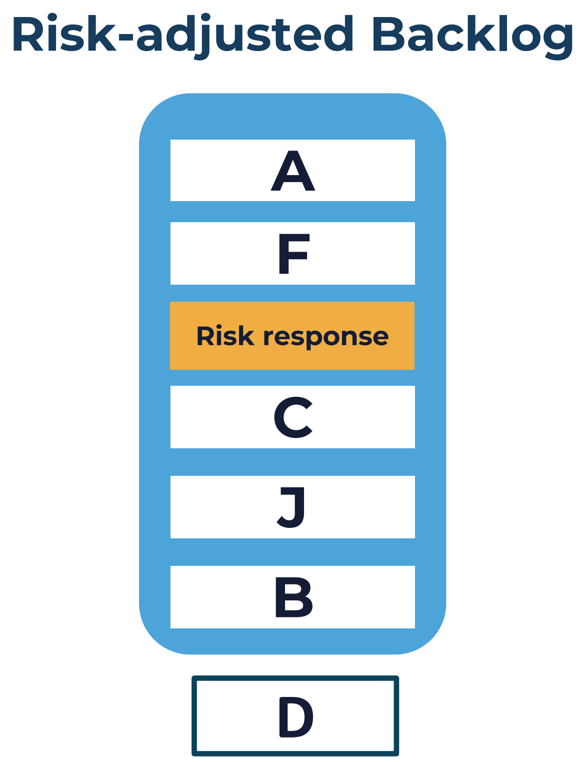
- **You have 1 hour and $25**
- **Does that change the order of the backlog and what you will do next?**

Risk-Adjusted Backlog

Based on Expected Monetary Value (EMV)

EMV = Probability x Impact





Expected Monetary Value

- Quantitative analysis of risk
- Expected Monetary Value (EMV)
  - EMV = Probability x Impact

Example:
Risk has a 25% probability of happening.
If it does, it will cost the project $5,000.
EMV = .25 x -$5,000 = -$1,250

How to Manage Regulatory Compliance Work

Ongoing with development

- Adapt to new regulations quickly
- May need to repeat tests of previous work (rework)

After product development

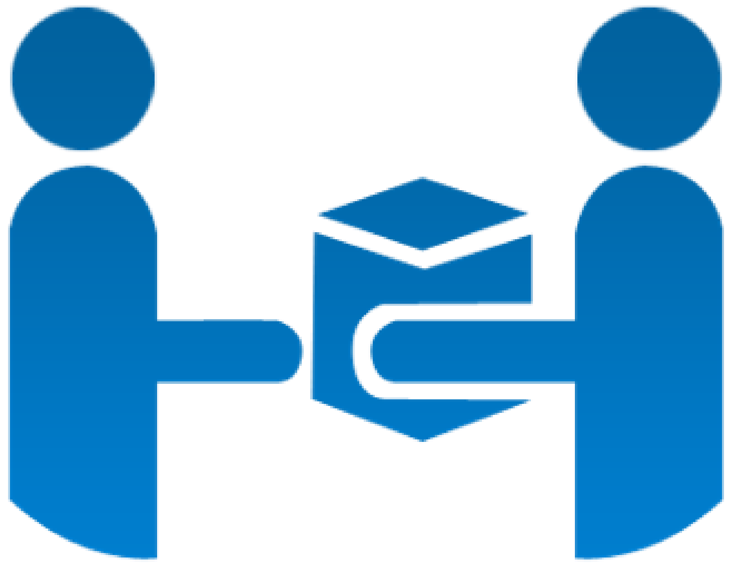
- Can’t react as quickly
- Prevents rework

Hybrid

- Test important components now
- Test entire product later
- Catch big issues

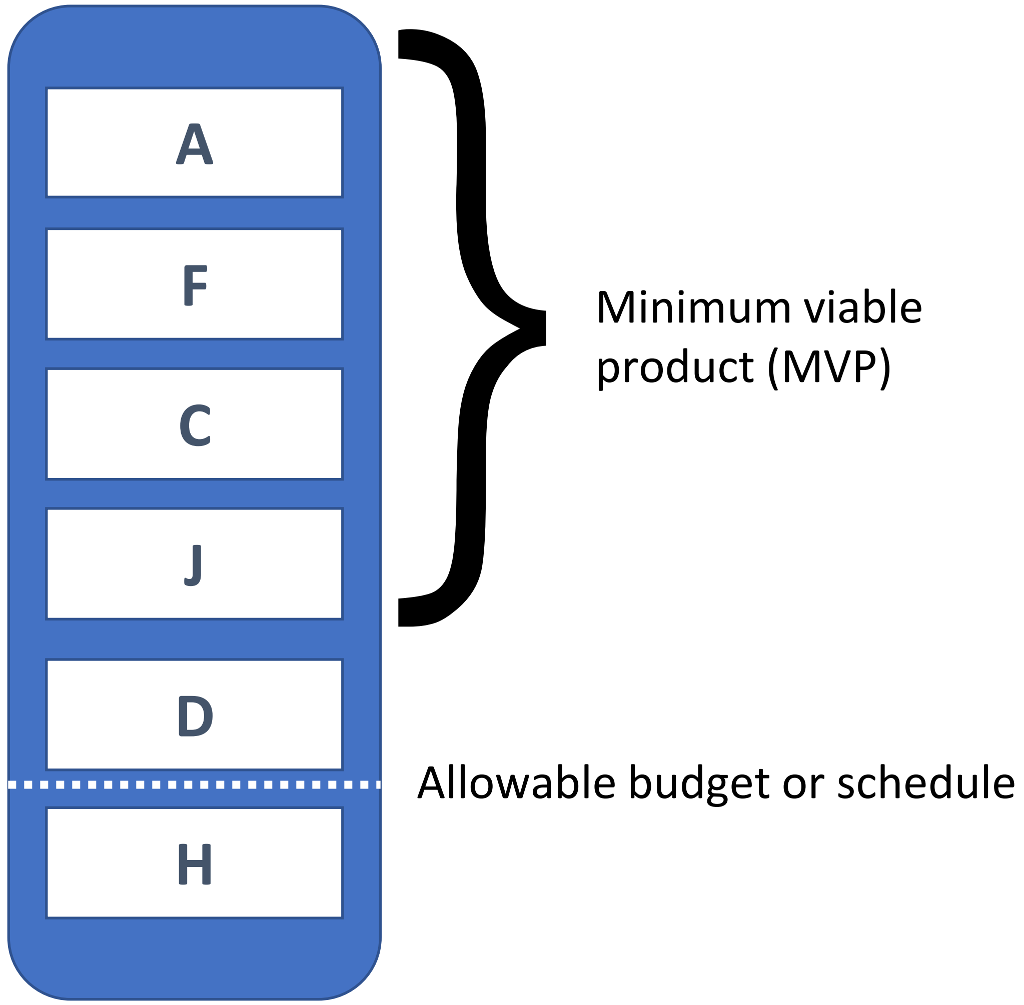
Minimum Viable Product (MVP)

- Also known as minimum marketable feature (MMF)
  - Complete enough to be useful
  - Small enough that it is not the entire project scope
  - Early release of MVP allows for rapid feedback and changes
  - Additional functionality can be included in future releases



Relative Prioritization

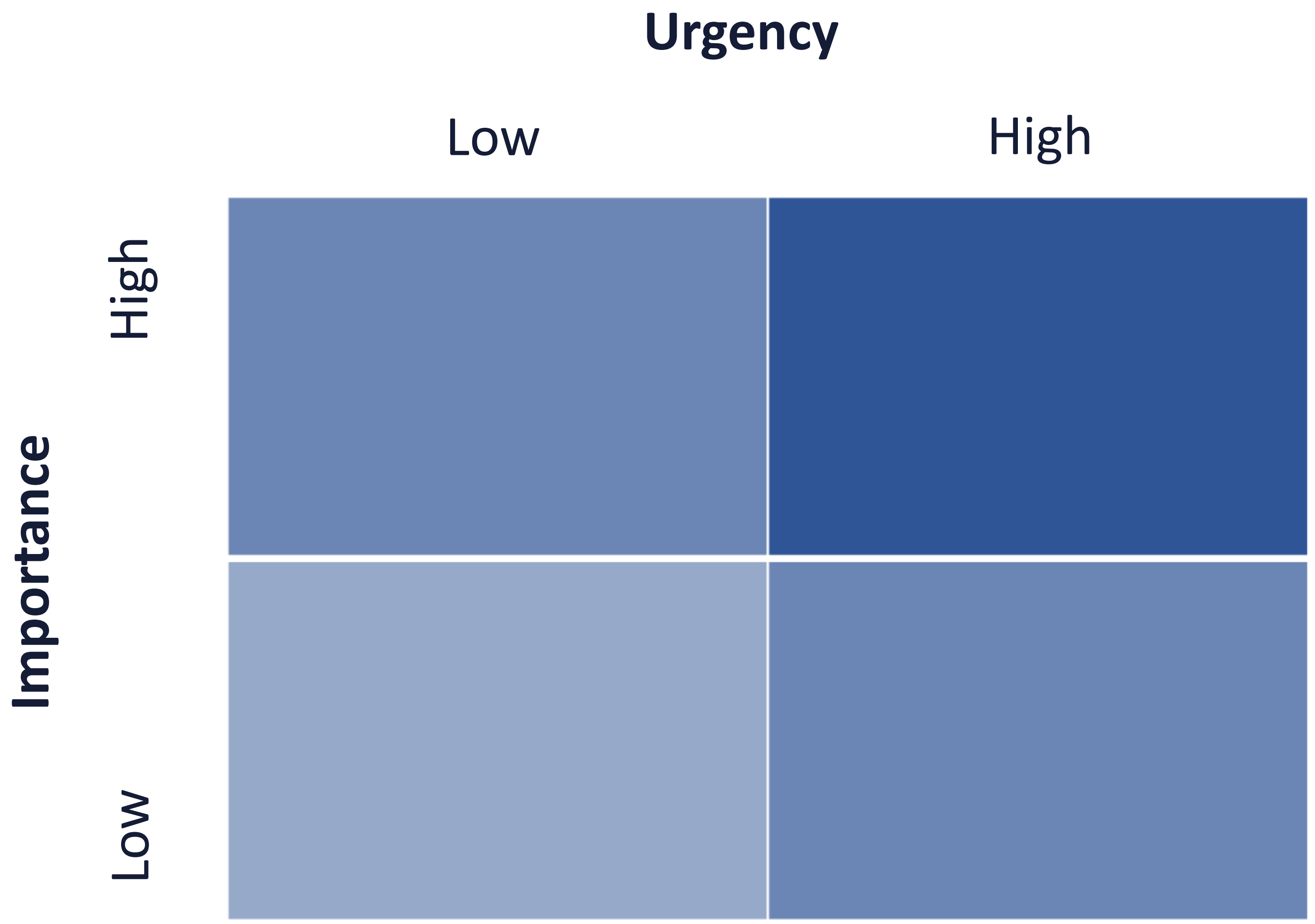
- Also known as relative ranking
- There are several techniques
  - Priority matrix
  - MoSCoW method
  - Monopoly money
  - Kano method
  - 100-point method
  - Dot voting/Multi-voting
  - CARVER technique



Priority Matrix

Can be tailored

- Value
- Cost
- Risk
- Complexity/ability to release



MoSCoW Method

Must have

o

Should have

Could have

o

Won’t have/would like to have

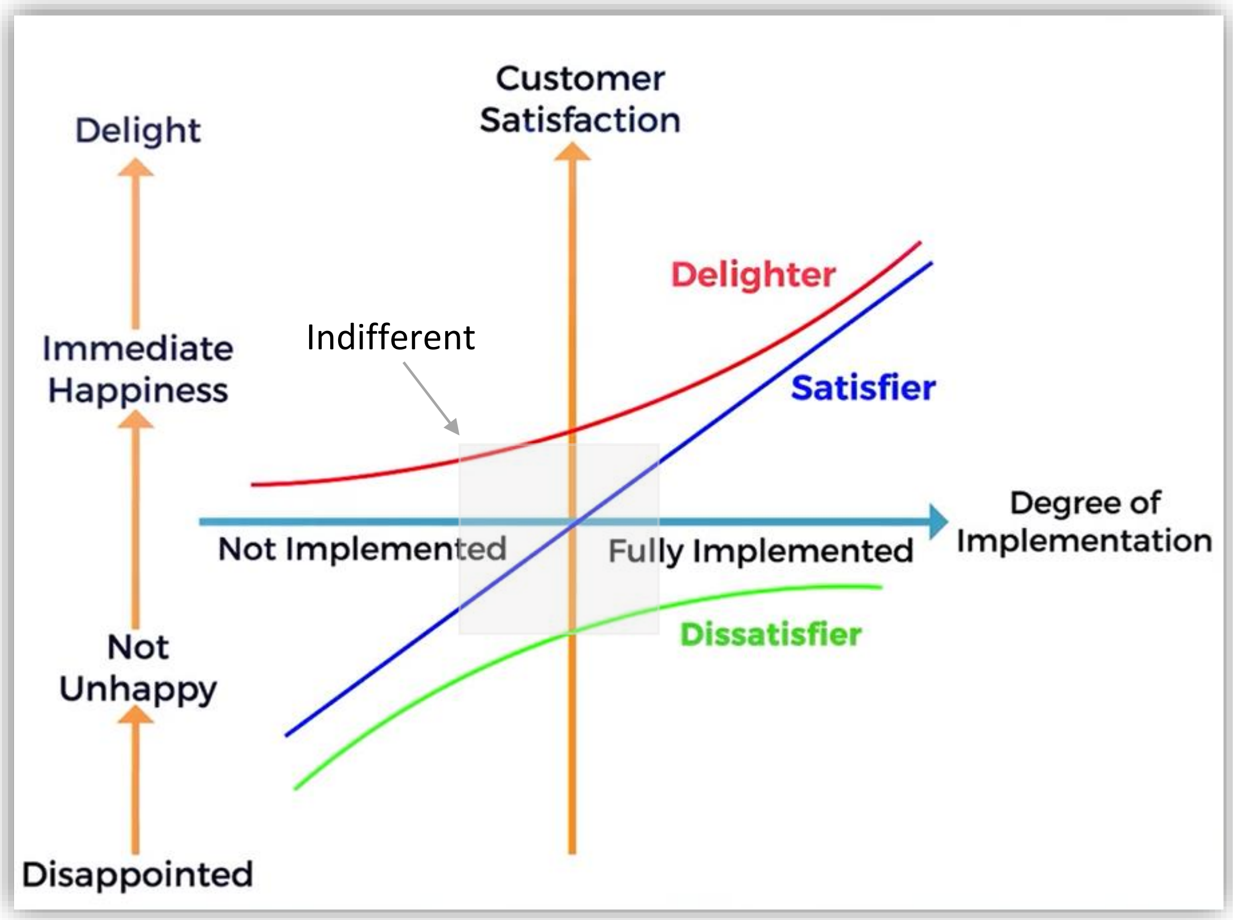
Category	User Stories
Must have	Included with the release
Should have	Not critical but still important
Could have	Useful and would add value
Won’t have	Excluded from this release
Would like to have	Retained for the future

Play Money

- Participants use money to “buy a feature”
- Features with the most money are the highest priority
- Feature prices may be set based on story points, hours of effort, or complexity

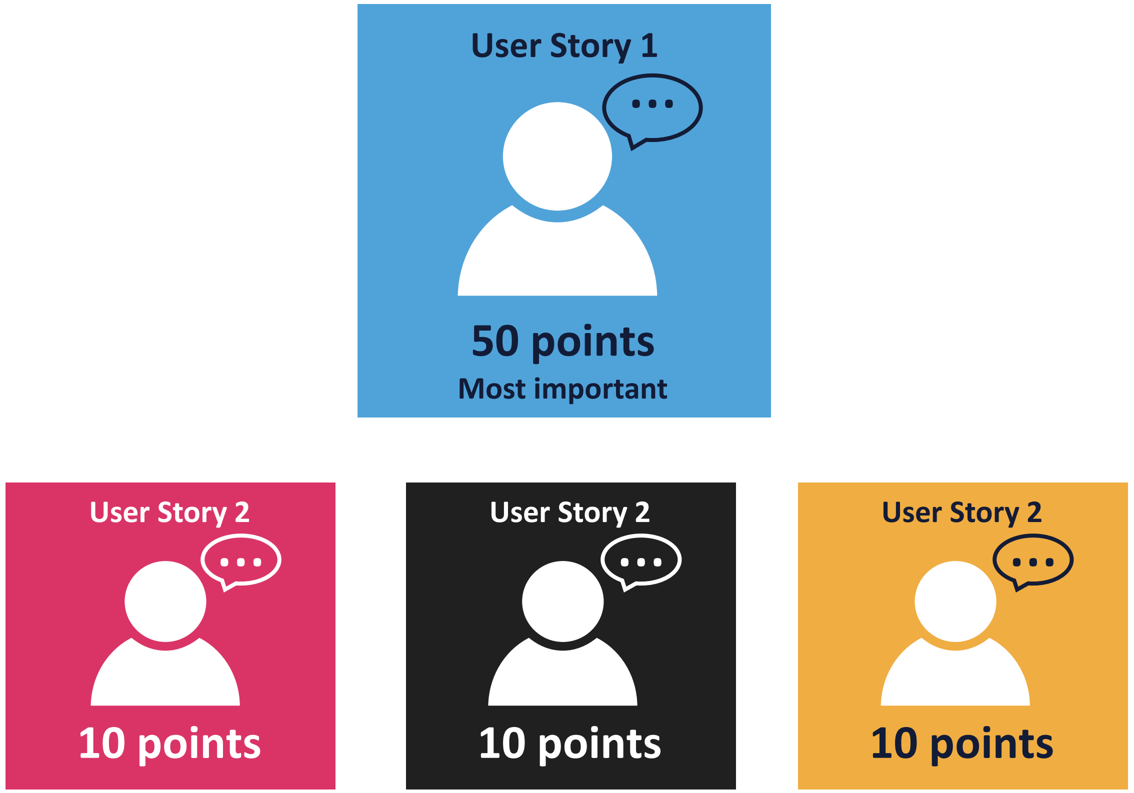
Kano Model

- Classify Customer Preferences
  - Delighters/Exciters
  - Satisfiers
  - Dissatisfiers
  - Non-essential/Indifferent



100-Point Method

- Each stakeholder has 100 points to spend on requirements
- The points can be allocated in any way
- Requirements are prioritized by points



The 100-point method was developed by Dean Leffingwell and Don Widrig for use cases.

Dot Voting or Multi-Voting

Follows brainstorming

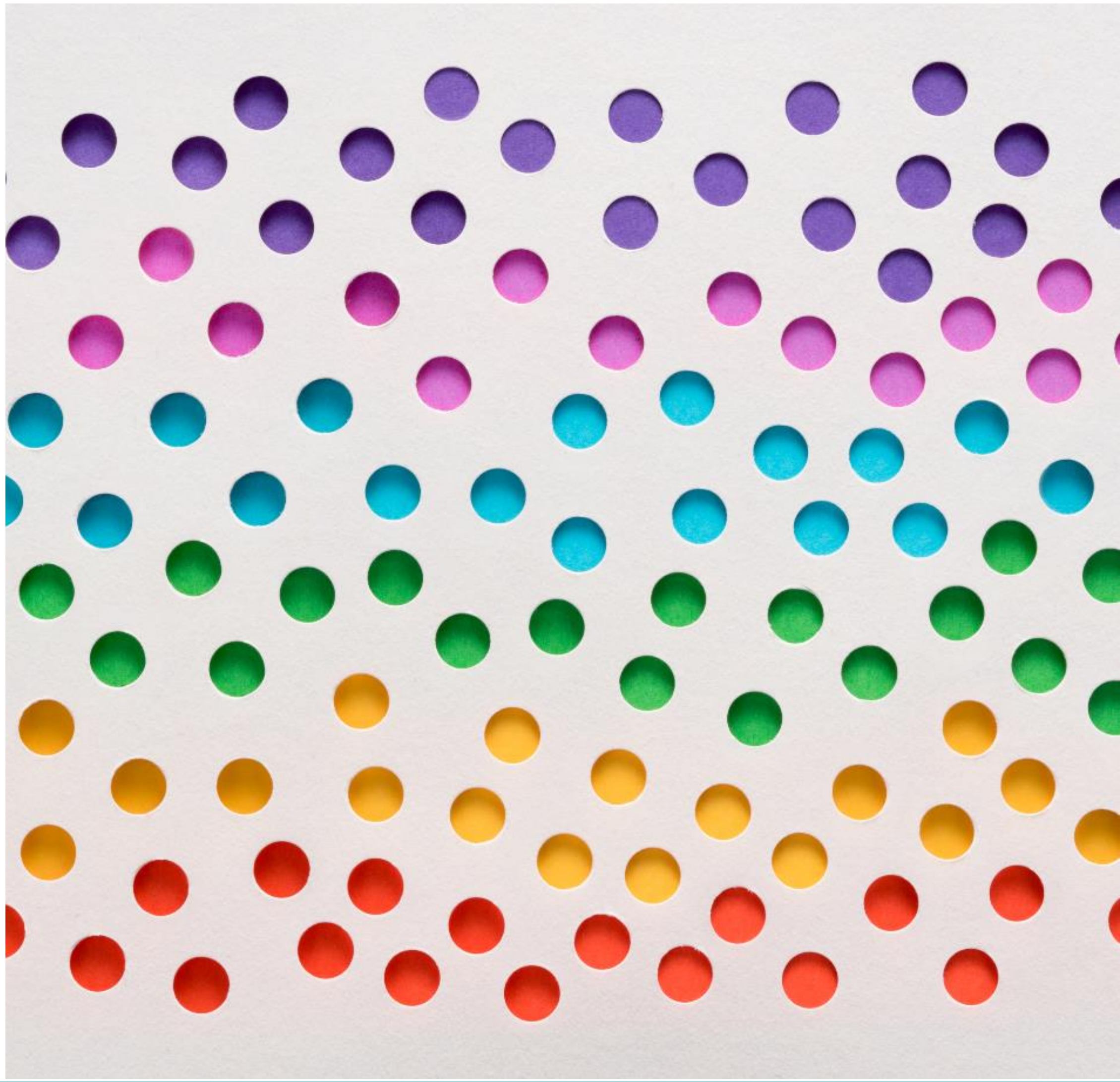
Each person can vote for 20% of the choices

Results show which features are valued by the most stakeholders

Example:

20 items must be prioritized

Each person gets 4 votes



CARVER Technique

Feature	Criticality	Accessibility	Return	Vulnerability	Effect	Recognizability	Total
Feature #1	1	4	4	5	2	5	21
Feature #2	3	2	3	3	3	3	17
Feature #3	5	1	3	2	4	1	16

- Categorizes user stories based on criteria
  - Criticality: is it absolutely necessary?
  - Accessibility: can we start on it now, or do we need something else first?
  - Return: does the value justify the cost?
  - Vulnerability: is it easy or difficult to achieve?
  - Effect: how does it impact the overall goal?
  - Recognizability: is it easy to understand?
- Visualize trends
- Prioritize requirements

Agile Estimating Techniques

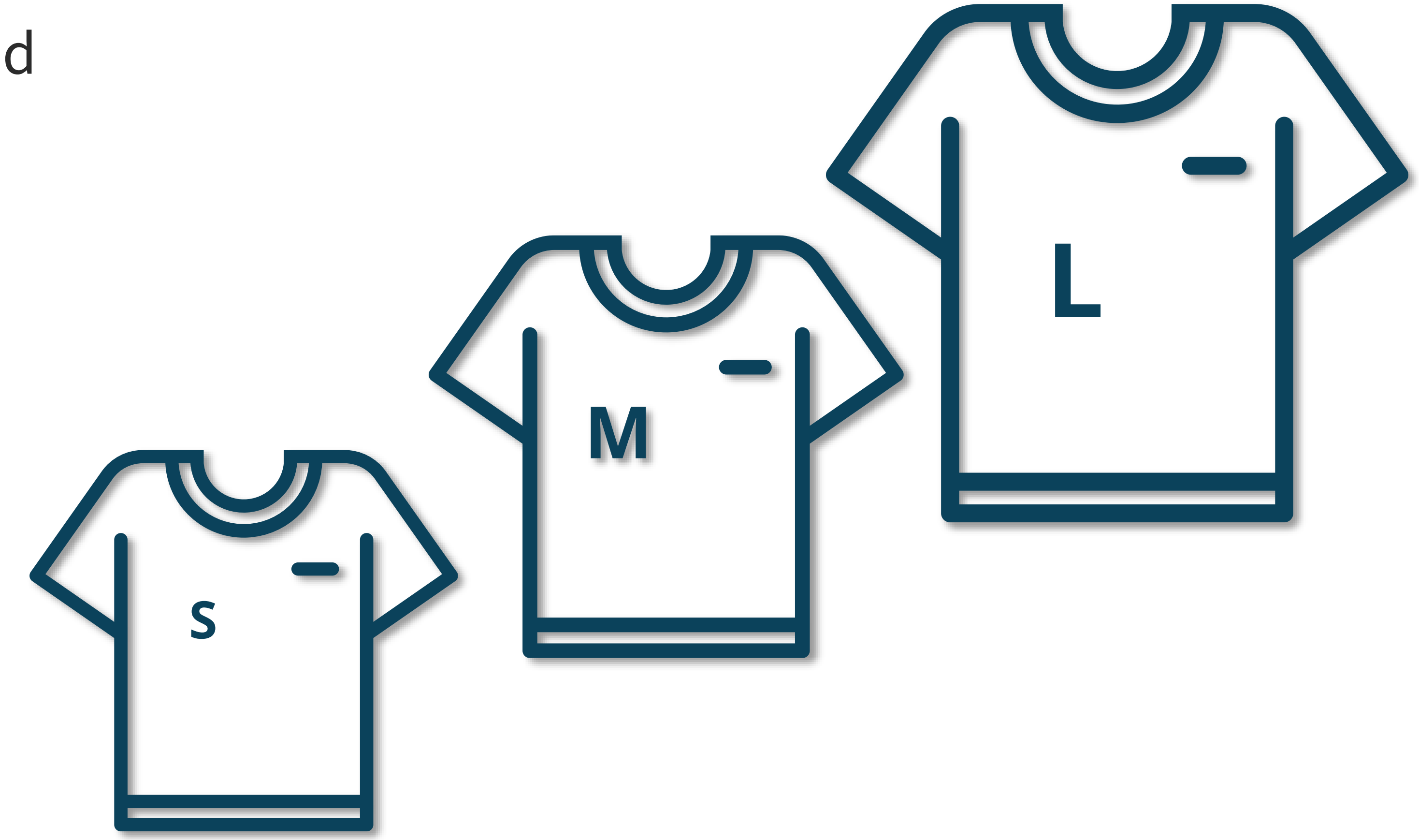
- Relative estimation
- Arbitrary measure
- Usually used by scrum teams
- Express effort required to implement a story
- 3 items taken into consideration: level of complexity, level of unknowns, effort to implement.

T-shirt sizing

Quick and easy technique

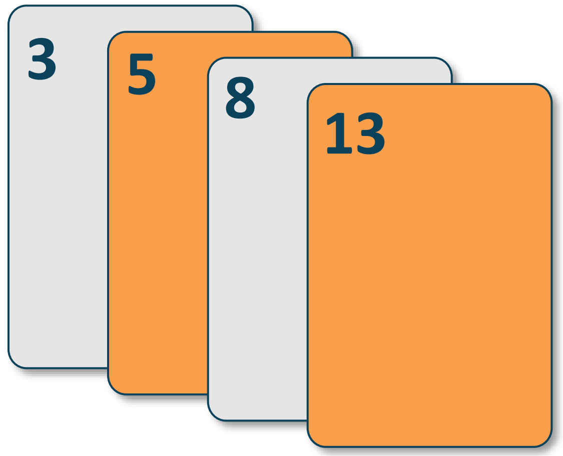
Absolute value not considered

Sizes instead of numbers



Story Points

- Relative sizing
  - We aren't good at absolute estimate
  - We are better at relative estimates
- Not tied to days, hours, or dates
  - Removes pressure or emotion
- Based on quantity of work, not speed
- Unique to a team
  - Not comparable to the work of other teams
  - Removes competition between teams
- Reference for future estimates
- Reserves and buffers are not necessary



While story points is the most commonly used metric, teams may choose any unit to represent work.

Video

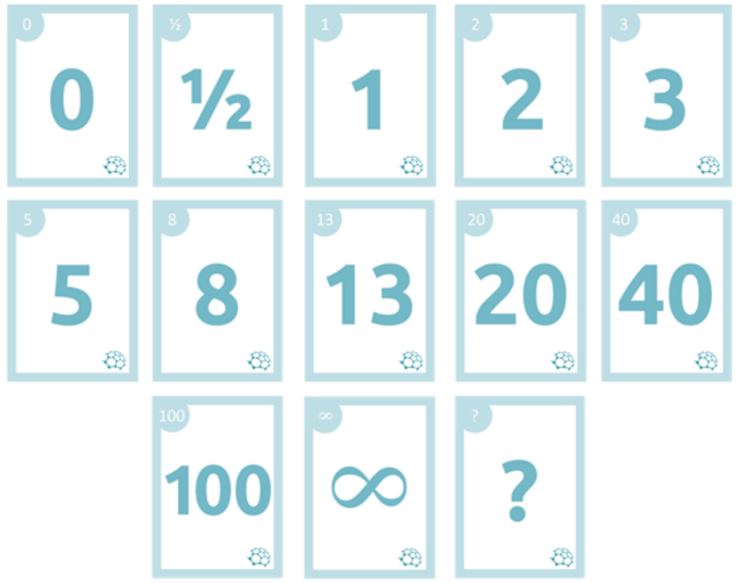
Agile Estimating and Planning: Planning Poker

5:31 run time

Mountain Goat Software
https://youtu.be/gE7srp2BzoM

Planning Poker

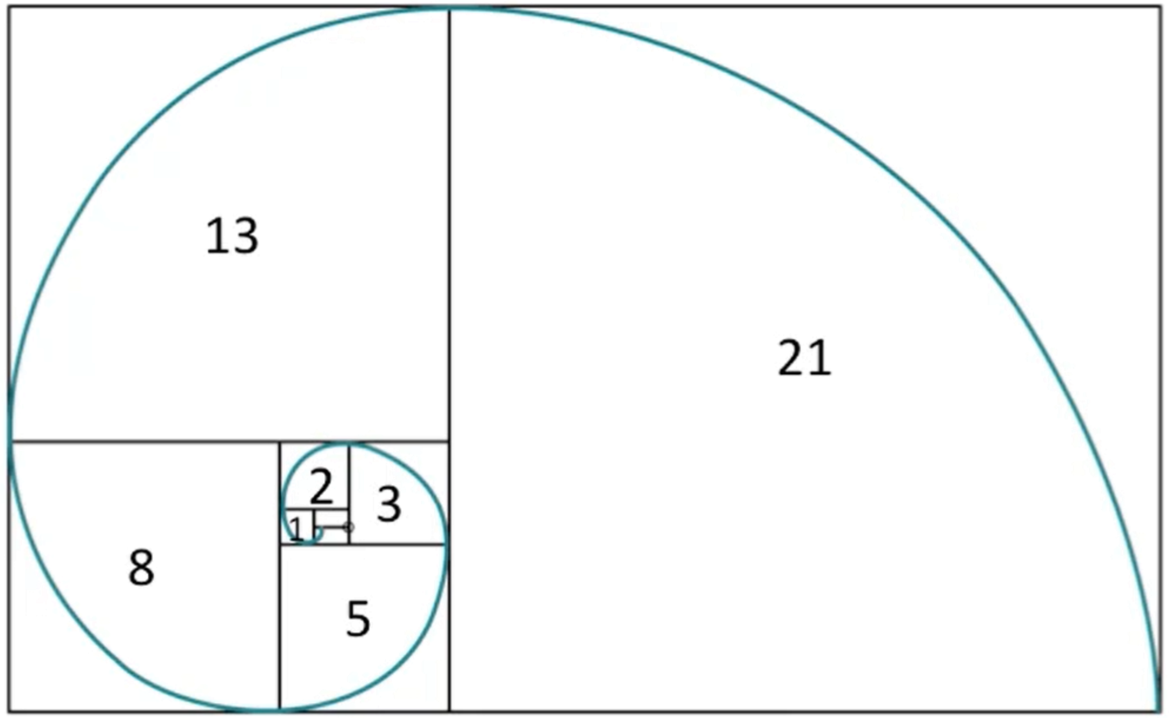
- Uses Fibonacci sequence
- Each player receives a deck of cards
    - Facilitator reads a user story
    - On the count of 3, everyone shows their estimate
    - Purpose is to build consensus
    - close to consensus, move on and round to higher number
    - Scattered estimates, discuss and estimate again
    - Estimates are approximates



Fibonacci Sequence

Sequence of numbers

- Used for estimating story sizes
- Each number is the sum of the two preceding numbers
- **0, 1, 1, 2, 3, 5, 8, 13, 21, 34**, and so on



Scaled Agile Framework® (SAFe) Prioritization Technique

Weighted Shortest Job First (WSJF)

Calculating the Cost of Delay:

WSJF = Cost of Delay/Job Size (Duration)

https://www.scaledagileframework.com/wsjf/

Scaled Agile Framework® (SAFe) Prioritization Technique

Weighted Shortest Job First (WSJF)

Calculating the Cost of Delay:

- Consider three variables
  - User-Business Value
  - Time Criticality
  - Risk Reduction/Opportunity Enablement
- Use the Fibonacci sequence to give each job a score: 1, 2, 3, 5, 8, 13, etc..
- Give the smallest job a 1 and use relative estimating to score the rest

https://www.scaledagileframework.com/wsjf/

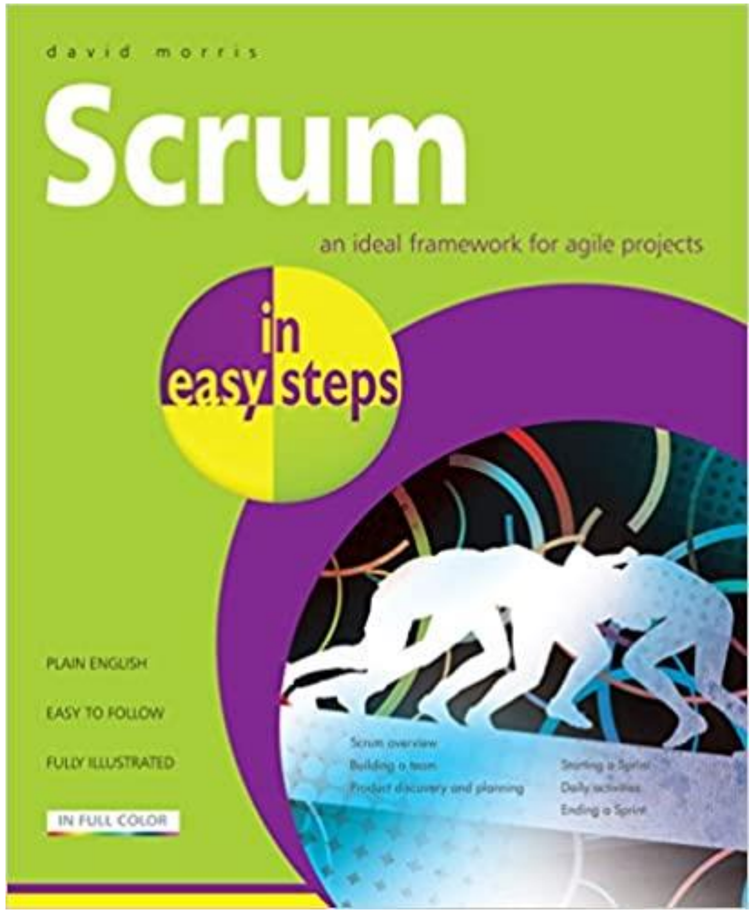
Video

Weighted Shortest Job First (WSJF)

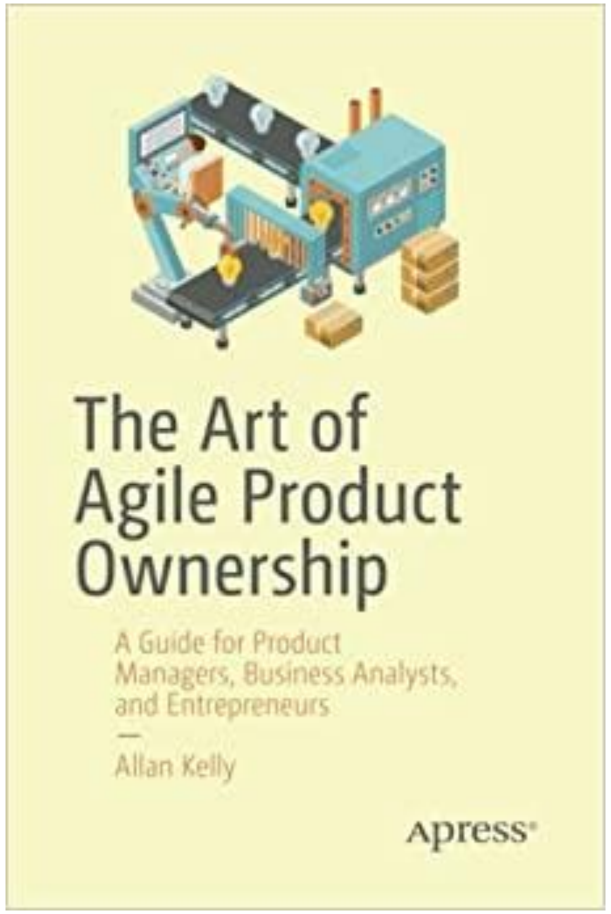
3:35 run time

Croesus
https://youtu.be/3PwJNCYZr8o

Recommended Reading: Percipio Books



Scrum in Easy Steps
Chapter 4: Defining the Product Backlog


The Art of Agile Product Ownership
A Guide for Product Managers, Business Analysts, and Entrepreneurs
Allan Kelly
Apress

The Art of Agile Product Ownership:
A Guide for Product Managers,
Business Analysts, and Entrepreneurs

Tool Earned!

Product Backlog