Project Management

Lecture 5

Ruzanna Chitchyan, Jon Bird, Pete Bennett
TAs: Alex Elwood, Alex Cockrean, Casper Wang

(Using materials created by N. Walkinshaw and R. Craggs)

Overview

- About Measurement
- Measurement under White Box:
  - Lines of code
  - Cyclomatic Complexity
- Measurement under Black Box:
  - Planning Poker
- Software Laws:
  - Patents, Copyright, Contract, Privacy

Measurement is Central to Quality

- How to plan for the project time and effort?
  - For the team?
  - For the customer?
- Which software/part of it needs more time for testing?
- Which developer should get a bonus payment for productivity?....

"You cannot **control** what you cannot measure."

Tom DeMarco, 1982

What is “Measurement”?

- Attributing values to objects.
  - The fuel efficiency of a car (gallons per mile)
  - The number of goals scored by a footballer
  - The cost of a house
- Can use these values as basis for comparison
  - What is the cheapest house?
  - Who is the best goal scorer?
- Can use these measurements and comparisons to **make better decisions**.
  - Which car should I buy (e.g., given five candidate cars)
  - Which striker should I put in my team?

Measurement is Difficult in Software Engineering

- Most entities are difficult to measure reliably
- Difficult or impossible to “pin down” a single value

E.g., Software Quality (ISO/IEC 25010):

- Functional Suitability
  - Functional Completeness
  - Functional Correctness
  - Functional Appropriateness
- Performance Efficiency
  - Time Behaviour
  - Resource Utilisation
  - Capacity
- Compatibility
  - Co-existence
  - Interoperability
- Usability
  - Appropriateness
  - Realisability
  - Learnability
  - Operability
  - User Error Protection
  - User Interface Aesthetics
  - Accessibility
- Reliability
  - Maturity
  - Availability
  - Fault Tolerance
  - Recoverability
- Security
  - Confidentiality
  - Integrity
  - Non-repudiation
  - Authenticity
  - Accountability
- Maintainability
  - Modularity
  - Reusability
  - Analysability
  - Modifiability
  - Testability
- Portability
  - Adaptability
  - Installability
  - Replaceability

Usual Metrics: Size and Complexity

- After development ...
  - How much effort will it require for maintenance?
  - Where should we direct testing effort?
  - How much effort was required for development?
  - Metrics are based upon source code (“white box”)

- Before development has started ...
  - How much programming effort will module X require?
  - What will be the estimated cost of the final product?
  - Metrics are based upon requirements / specification (“black box”)

White Box Complexity Metrics

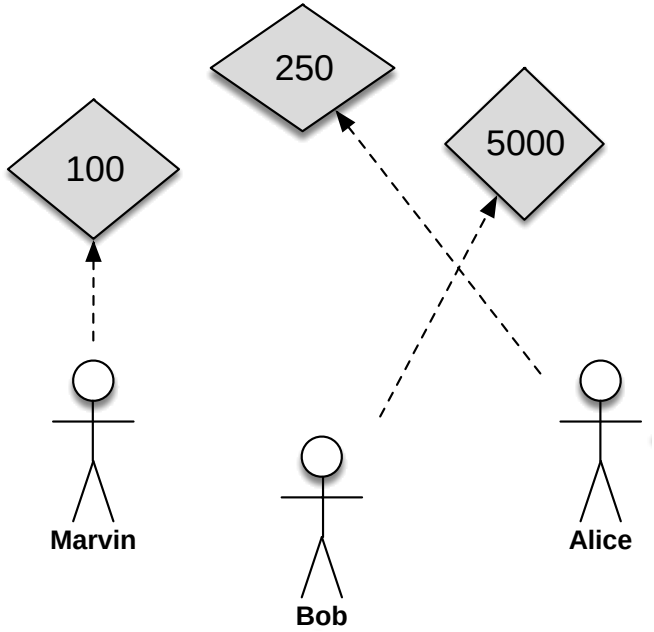
Number of lines in a file (or a group of files)

- Easy to compute
- Easy to understand and interpret
- Often sufficient for an approximate measure of size
- Widely used (perhaps the most widely used) metric

- Comments
- What is a line?
- Blank lines
- Not all “lines” are equal
- Ignores logical/ architectural complexity
- Highly language-specific

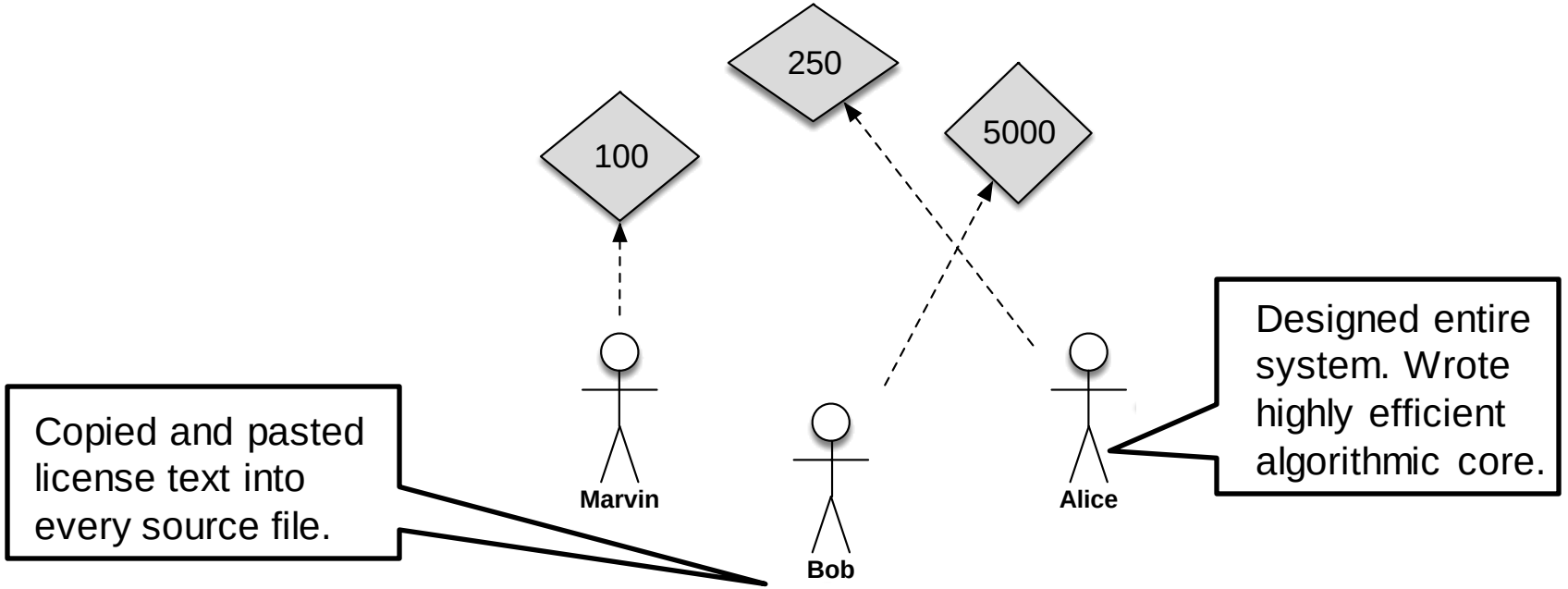
Example: Who is the most productive programmer?

Measured in lines of code



Example: Who is the most productive programmer?

Measured in lines of code



Cyclomatic Complexity

- Calculated from the control flow graph:

  V(G) = E – N + 2P

  E – number of edges;

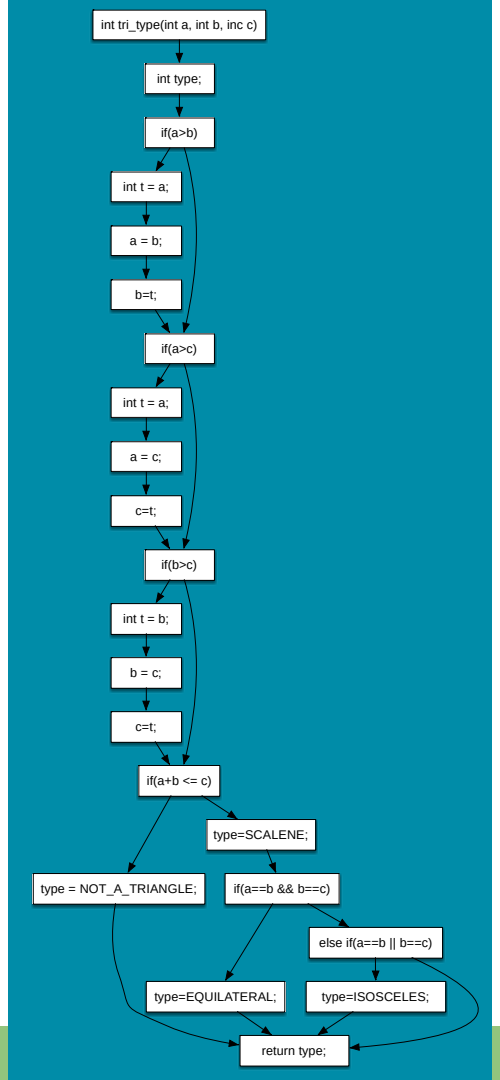
  N – number of nodes;

  P – number of procedures (usually 1)

- Number of independent paths through the code
- Independent path – any path that introduces at least one new statement/condition

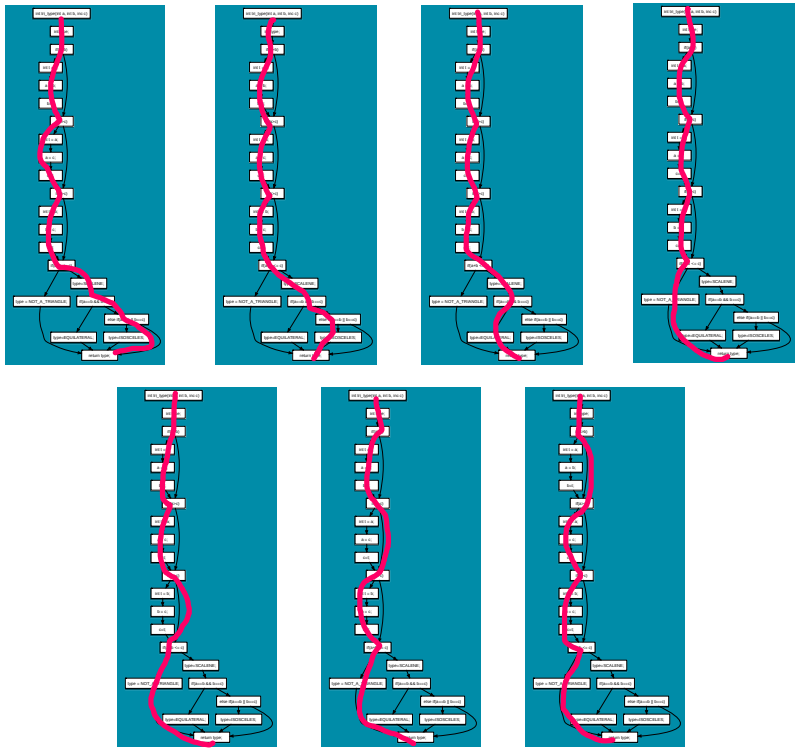
Triangle Example

```
1   int tri_type(int a, int b, int c) {
2     int type;
3     if (a > b)
4       { int t = a; a = b; b = t; }
5     if (a > c)
6       { int t = a; a = c; c = t; }
7     if (b > c)
8       { int t = b; b = c; c = t; }
9     if (a + b <= c)
10      type = NOT_A_TRIANGLE;
11    else {
12      type = SCALENE;
13      if (a == b && b == c)
14        type = EQUILATERAL;
15      else if (a == b || b == c)
16        type = ISOSCELES;
17    }
18  return type;
19  }
```



Number of Edges = 27
Number of Nodes = 22

V = 27 – 22 + 2 = 7



Black Box Complexity Merics

Estimating Agile Projects

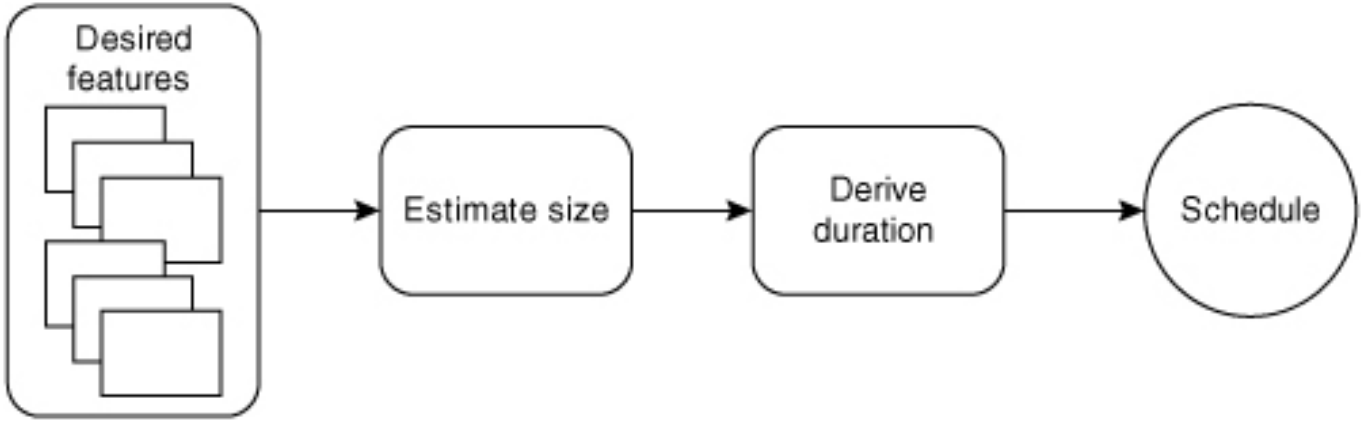


Figure from: Agile Estimating and Planning by Mike Cohn

Storey Points (Size Estimation)

- An informal, agile unit of “size measurement”
    - Usually an estimate from 1-10
- Derive an estimate from the whole team at sprint planning meetings
- Based on the idea of the “Wisdom of the Crowds”
    - The collective estimate of groups (i.e., of effort required for a story) is better than the estimate of an individual

Accuracy vs Effort in Project Estimation

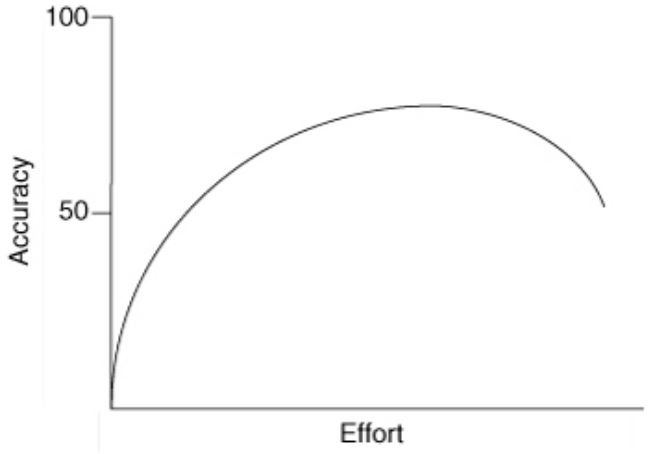
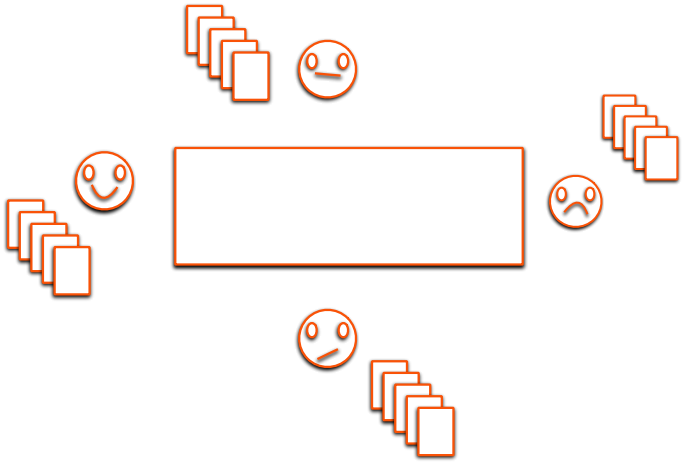


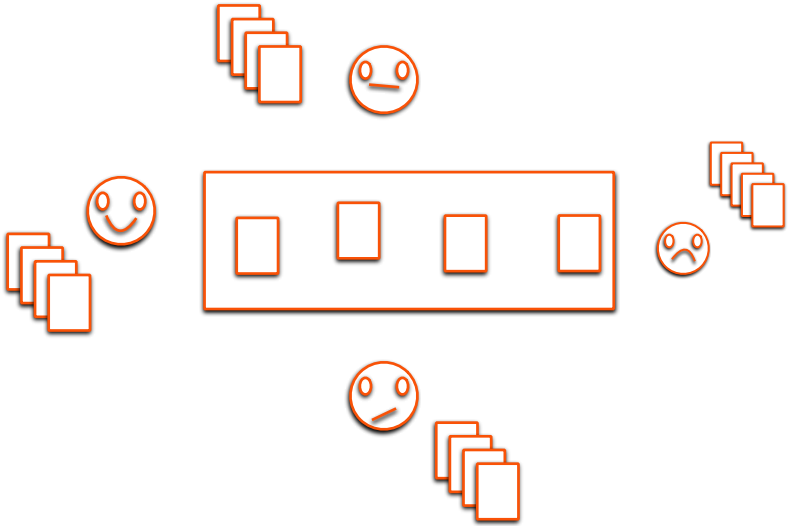
Figure from: Agile Estimating and Planning by Mike Cohn

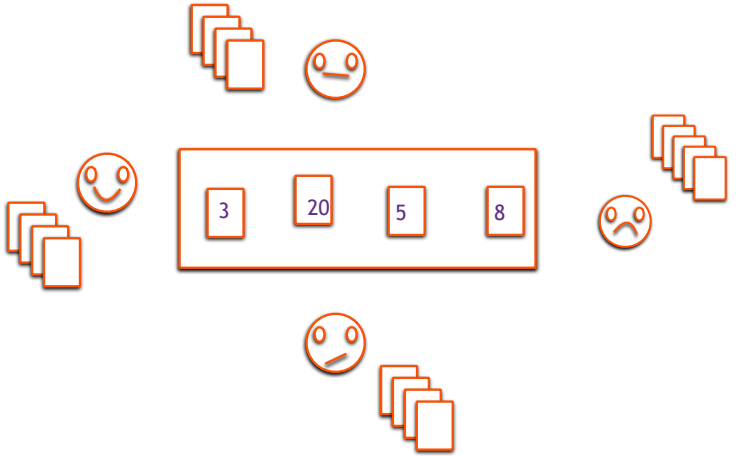
Planning Poker

- The whole team is involved
- Each member is given a set of numbered cards
- Numbers follow the Fibonacci sequence
- 1,3,5,8,13,20,...
  - Larger tasks become harder to estimate in exact terms
  - Low values - trivial to implement
  - High values - difficult to implement
- Each member is also given a “?” card

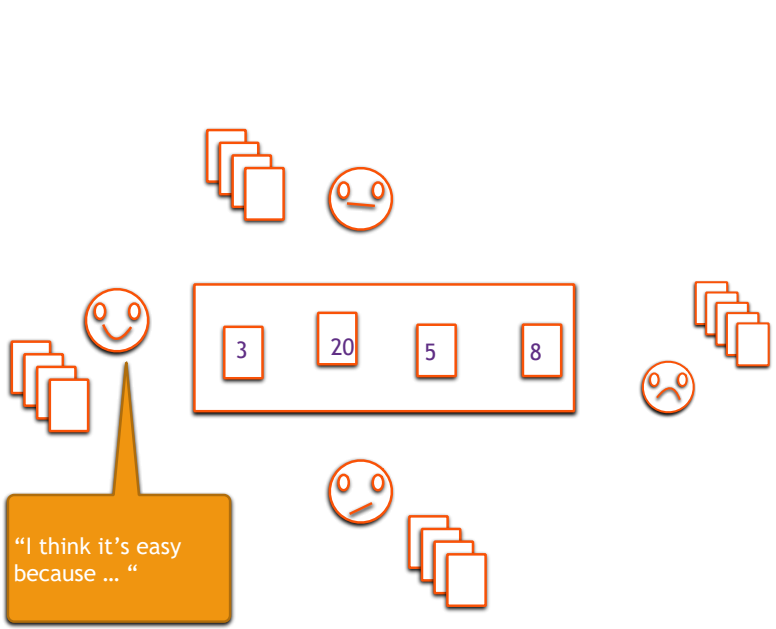


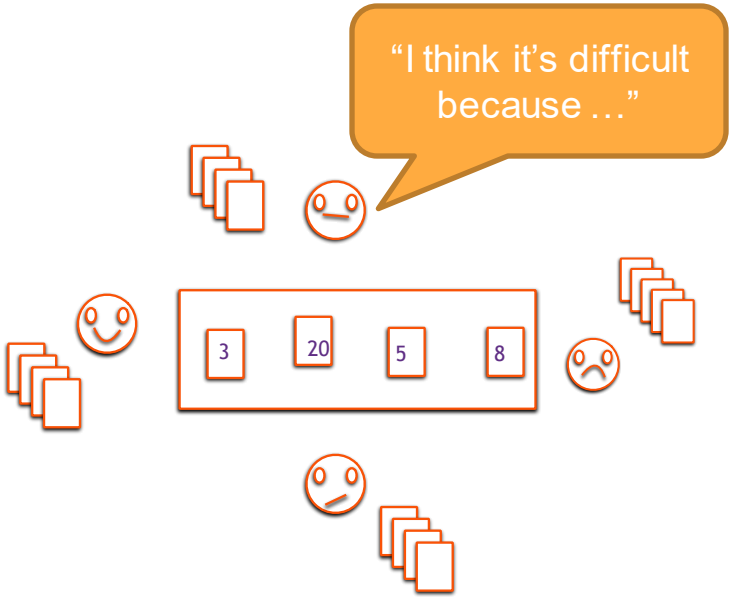
Planning Poker: Process





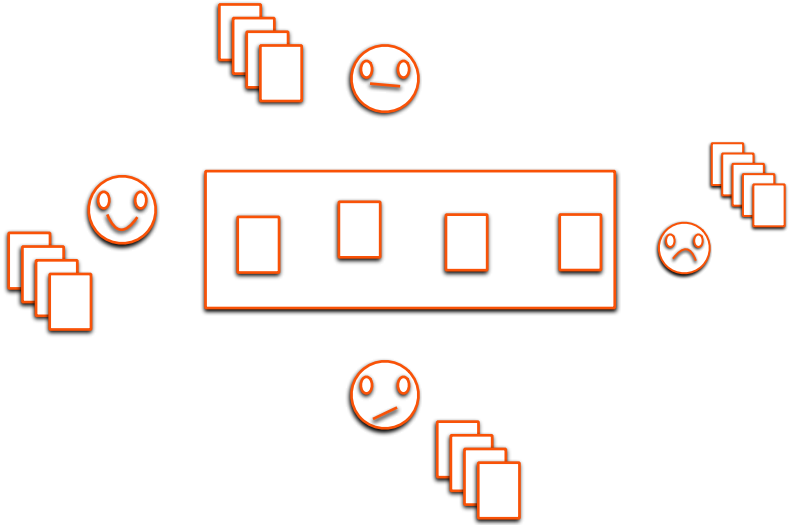
Planning Poker: Process

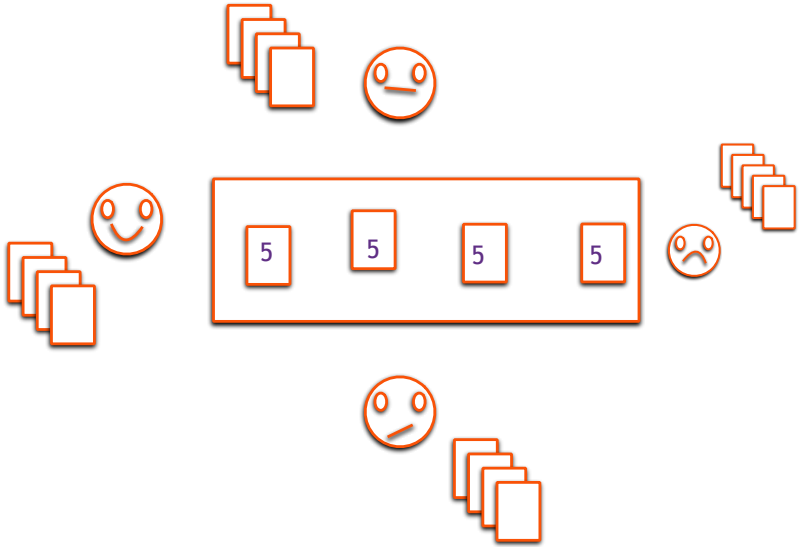




Planning Poker: Process

Cycle repeats for a maximum of 3 iterations (to avoid infinite loops!)

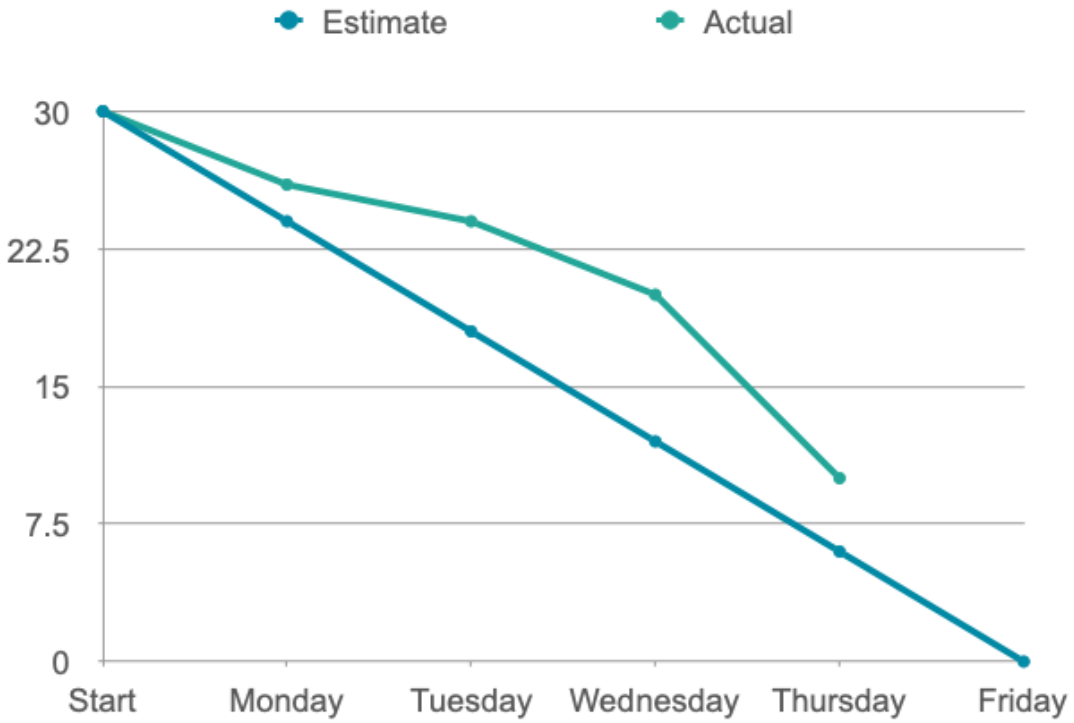




Team Velocity

- Number of (estimated) story points implemented per sprint.
- Can be derived from previous sprints.
    - e.g., Average points implemented from previous *x* sprints.
- Can be used to estimate:
    - Time required to complete project.
    - Target number of stories that can be completed in a sprint.

Burn Charts


Estimate
Actual
30
22.5
15
7.5
0
Start
Monday
Tuesday
Wednesday
Thursday
Friday

Software Laws: Patents, Copyright, Contract, Privacy

Patent Law

A government license giving a right for a set period, especially to exclude others from making, using, or selling an invention

- Granted by the government
- to stop others exploiting **your** invention
- Lasts 20 Year

Inventions Must

- be **new**
- be an **inventive** step (not an obvious improvement)
- capable of industrial application

The “Social Network”

Did Mark Zuckerberg infringe a patent?

- No patent was granted
- The idea was not new, social networks existed before this

Copyright

- Creator has **exclusive rights** to perform, copy, adapt their work.
- Everyone else must get **Permission** (and possibly pay)
- "literary, dramatic, musical and artistic works" **includes software**
- Automatically owned (not granted)
- Lasts **70 years** after authors death (lots of exceptions)

This affects software in 2 different ways:

- Illegal Copies of Applications (Piracy) !
- Using someone else's code/UI design/etc. in your application

(Not the "idea" but the actual "stuff" (code, design, documents) created by someone else)

Copyright Theft?

NO:

- Get permission (obtain a licence)
- Be within "fair use" (e.g. for study or review)
- Use "open source" software
- Create something similar yourself, independently
- "Obvious" code can't be copywrited

YES:

- Displaying an image from another page
- Using code found on the internet
- Copying Windows 95 for your friends

The “Social Network”

Did Mark Zuckerberg infringe copyright?

Maybe

- but there is no evidence he copied
- it it's not fair use
- it wasn't OSS
- he saw the code so didn't invent it himself

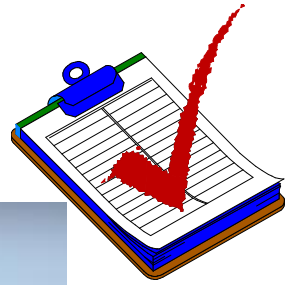
Contract Law

Employer contracts usually force an employee to:

- Not work for anyone else
- Hand over any ideas (Intellectual Property)
- Not disclose company secrets (Non-disclosure-agreements)

  (even after you stop working for them)

The “Social Network”



Did Mark Zuckerberg break contract?

Probably Not

- there was no written contract
- he did not disclose any secrets about the other project

Data Protection

- **UK** : Data Protection Act
- **EU** : Data Protection Directive
- **US** : a "patchwork" of state and national laws

8 Principles of Data Protection:

Any company storing "personal data" must make sure it is:

- fairly and lawfully processed (consent, contractual and legal obligations, public interest, ...)
- processed for **limited purposes**;
- adequate, relevant and **not excessive**;
- accurate and, where necessary, kept up to date;
- not kept longer than necessary;
- processed in accordance with the data subject's rights;
- secure;

not transferred to countries without adequate protection

Review

- How can we measure complexity?
- Why do we use black box options?
- What is a patent
- What is the difference between patent and copyright?
- What do we learn about contract from Social Network?