FMI

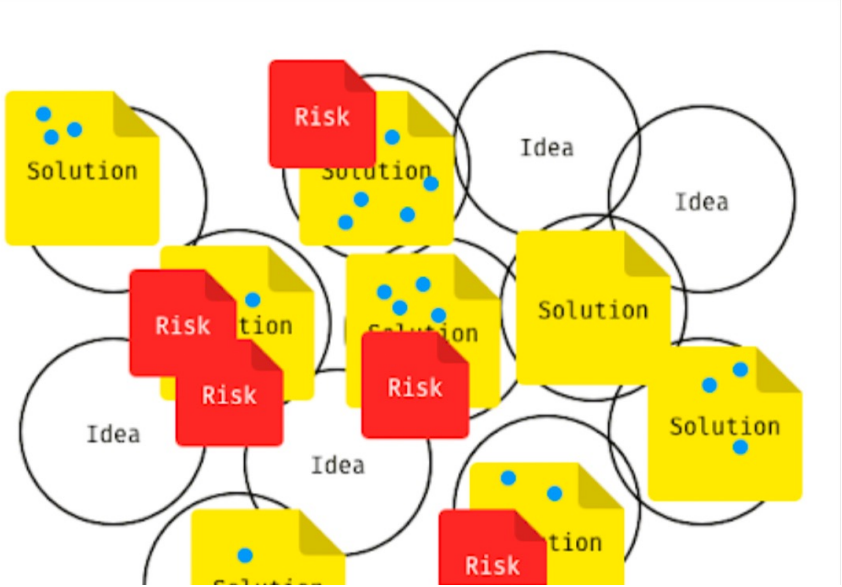
Part 2

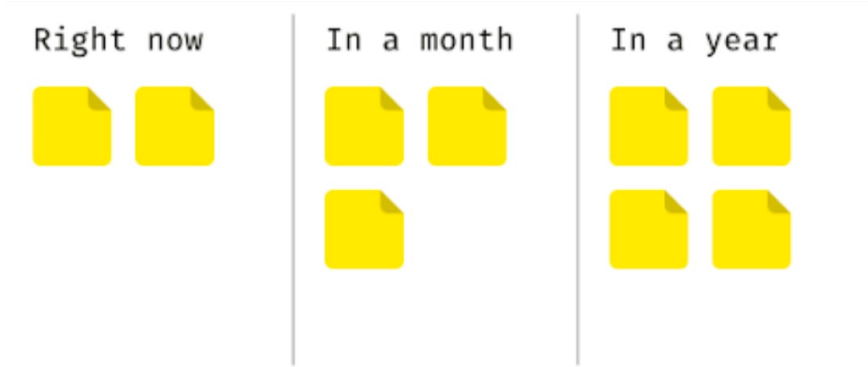
/thoughtworks

From idea to software

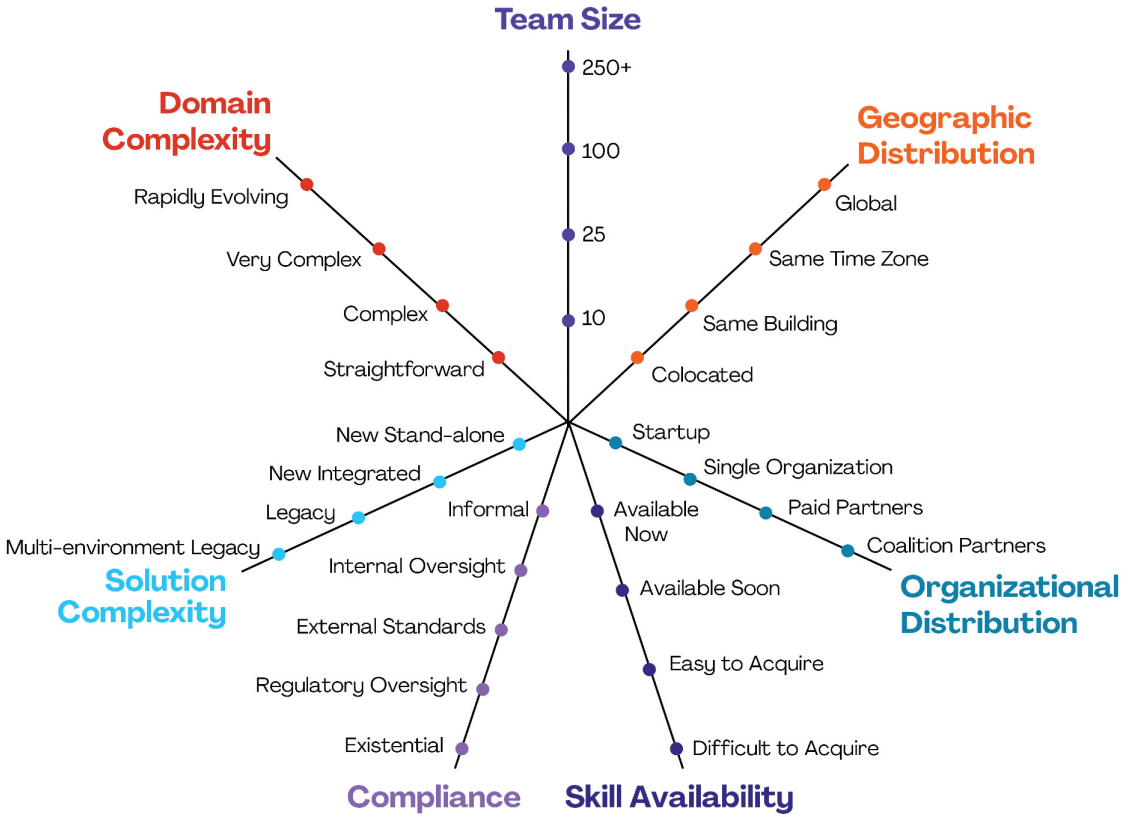
- Idea, brainstorming and roadmap
- Wireframes or mockups
- Proof of Concept (POC)
- Minimum Viable Product (MVP)
- Full Version 1 (Beta) Release
- The secret to success: **ITERATE**

Idea, brainstorming, and roadmap



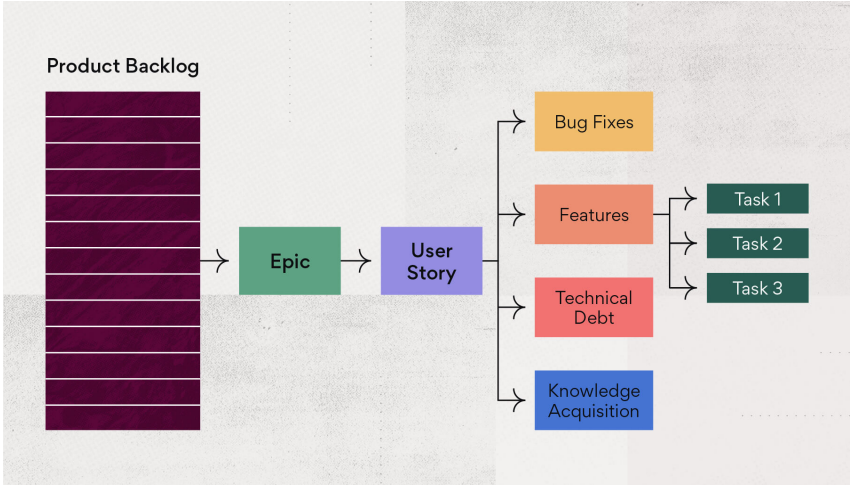


Context counts



Typical backlog

- Epics
- User stories
- Non-functional requirements
- Tasks
- Defects



Prioritizing the backlog: MoSCoW

- **Mo**st vital feature (aka must haves) – a feature without which you don't have a product or cannot live without
- **S**hould have – as important, but not vital
- **Co**uld have – desirable but not necessary, the "nice-to-haves"
- **W**on't have – may be considered for the future

Backlogs examples from previous years with FMI

- Beer Finder: https://www.pivotaltracker.com/projects/2315099
- Next Course Finder: https://www.pivotaltracker.com/n/projects/2315461
- Parking Finder: https://www.pivotaltracker.com/n/projects/1978201
- STB Finder: https://www.pivotaltracker.com/n/projects/1978171

User story definition

- What is a user story ?

  A user story describes functionality that will be valuable to either a user or purchaser of a system or software.

- Aspects: Ron Jeffries Three Cs

Card	***A written description of the story used for planning and as a reminder.***
Conversation	***Conversations about the story that serve to flesh out the details of the story.***
Confirmation	***Acceptance Tests to confirm that the story was coded correctly***

Source: XP Magazine 8/30/01, Ron Jeffries.

Why user stories

Emphasize verbal communication

Can be understood equally by developers and customers

Can be used for planning iterations

Support and encourage iterative development

Support participatory design

David Hussman on Helping Organizations Adopt Agile – User Stories
http://www.infoq.com/interviews/interview-david-hussman-agile

Examples

Generic user stories

As a user, I want to reserve a hotel room

As a user, I want to cancel a reservation

Specific user stories

As a vacation planner, I want to see photos of the room

As a frequent flyer, I want to rebook a past trip, so that I save time booking

- Identify clearly the different users and their expectations …
- … especially for primary users (work on the right priorities)

Excelent user stories – Gherkin

As a <user type>

I wan to <action>

So that <outcome>

Acceptance Criteria
Given .....
When .....
Then ....

Estimates
Assets (designs, APIs, signatures, mocks, test data etc.)

```
Feature: Some terse yet descriptive text of what is desired
  In order to realize a named business value
  As an explicit system actor
  I want to gain some beneficial outcome which furthers the goal

  Scenario: Some determinable business situation
    Given some precondition
      And some other precondition
     When some action by the actor
      And some other action
      And yet another action
     Then some testable outcome is achieved
      And something else we can check happens too

  Scenario: A different situation
      ...
```

INVEST in good user stories

- Independent – Avoid introducing dependencies (leads to difficulty prioritizing and planning)
- Negotiable – Stories are not written contracts and not all details are relevant
- Valuable – Valuable to users and customers
- Estimable – Because stories are used in planning
- Small – Large stories are hard to estimate
- Testable – Tests demonstrate that a story meets the customer's expectations

So a user story...

- Is a basic unit of work in an agile project
- Describes a desired piece of business functionality
- Is small enough to be implemented in an iteration
- A good user story is the simplest statement about the system that
  - The customer cares about
  - Test cases can be written to verify
  - Can be reasonably estimated
  - Can be reasonably prioritized

Definition of estimate

Estimate

- To form an **opinion** about
- A **tentative** evaluation or **rough** calculation
- A statement of the **approximate** cost of work to be done

Accurate

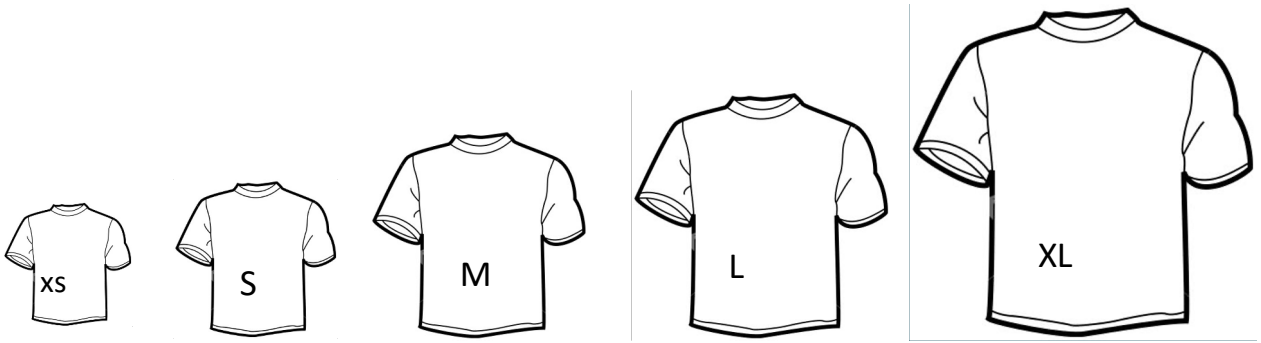
- Conformity to **fact**
- **Precision**
- **Exactness**
- Representation of the **truth**

Estimate ≠ Accurate

What to consider when estimating

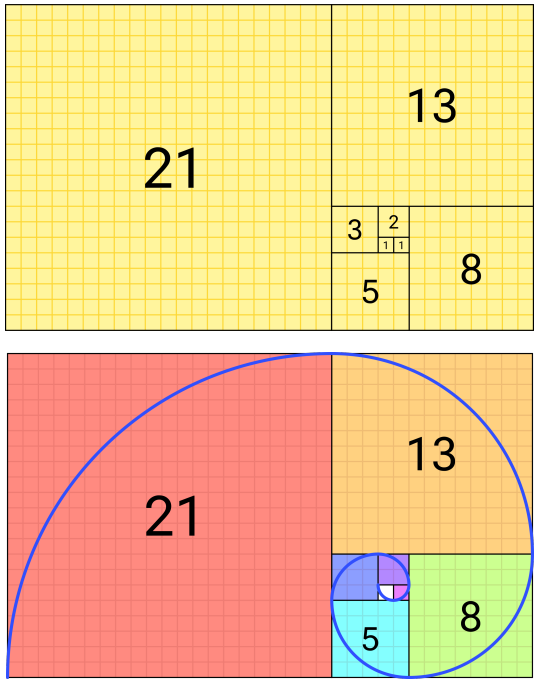
- Tests
  Unit test for each line, acceptance test for each story
- Refactoring
  The first-time duplication or inefficiency is a concern
- Infrastructure
  Considered for the first implementation of each layer
- Complexity
  Some tasks are tough, and need more time (unknowns?)
- Tedium
  Some tasks just take long because they do
- Everything
  Roll up all considerations into a single unit of measure

T-shirt sizes



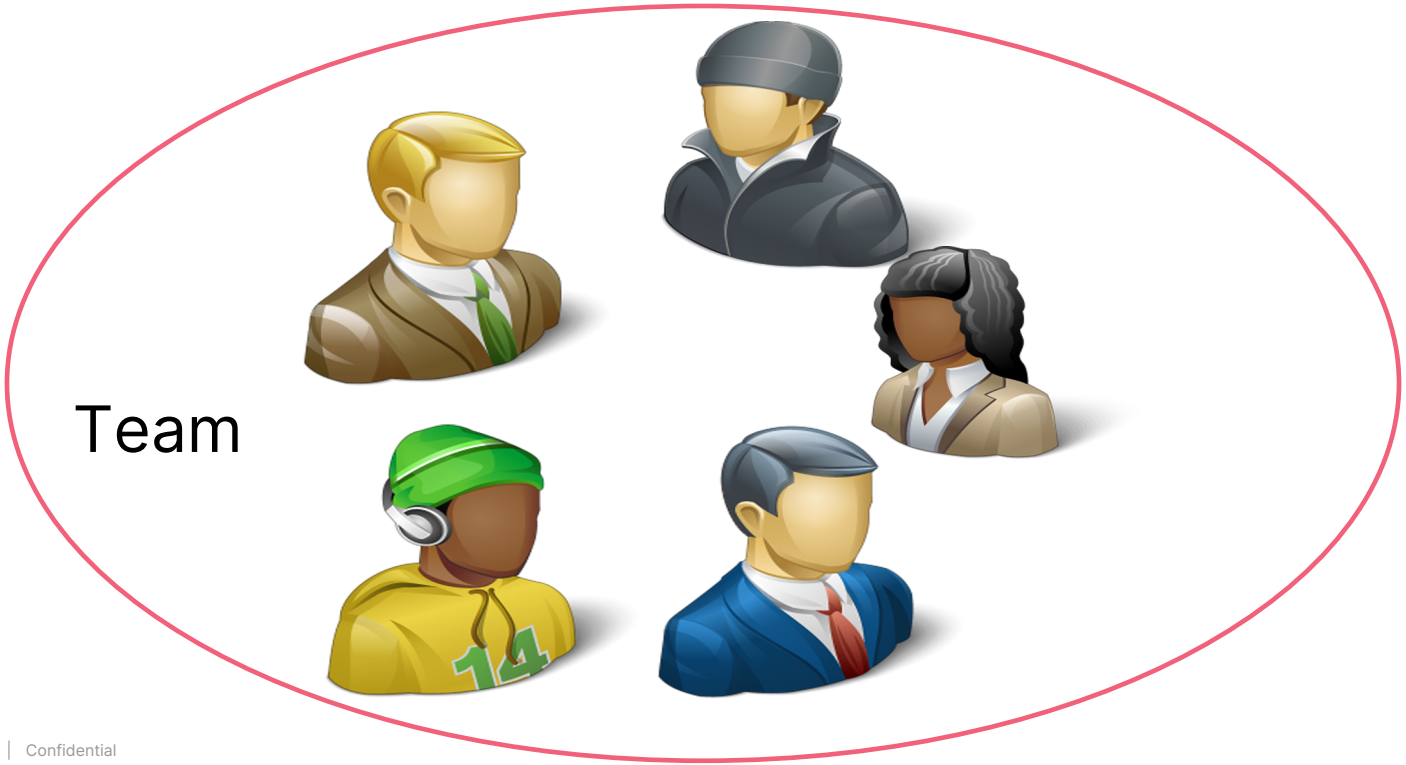
XS → Trivial, almost no effort and is very well known
S → Low complexity and effort, known and well understood
M → More complexity and/or more effort than a Small
L → High complexity and/or effort. These stories may be broken down closer to the time they are played
XL → Too Big. An epic story with lots of unknown. These stories need more attention.

Fibonacci



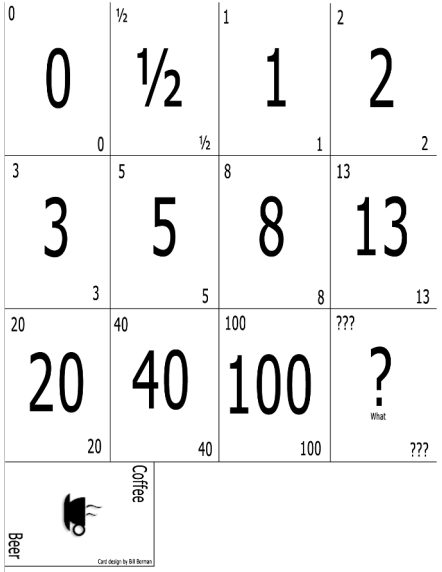
	Why	Story Points
Tasks	Too granular size, a sign for overdetailing	1
		2
Stories	Our targeted zone, right size for accurate sizing	3
		5
		8
		13
EPICS	Border on big, too much hidden complexity	21
	Very big, must split	55
	Huge size, must split even for sizing	144

Who estimates?



One way to estimate: Planning Poker

Planning Poker = Fibonacci + Playing Cards



Planning poker rules

- Product Owner / BA explains story
- Team discusses work involved
- Everyone estimates individually
- Everyone reveals estimates simultaneously
- Lowest and highest estimates are justified
- Repeat until estimates converge

Overconfidence in estimation

- 90-90 rule
  The first 90 percent of the code accounts for the first 90 percent of the development time. The remaining 10 percent of the code accounts for the other 90 percent of the development time.

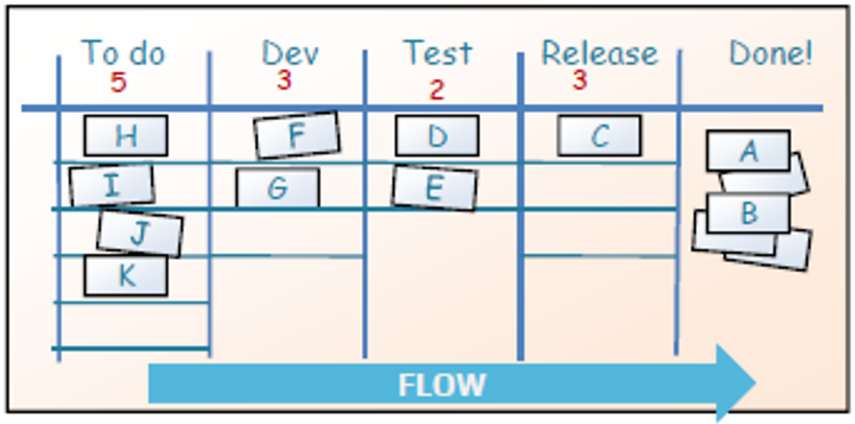
  – Tom Cargill, Bell Labs

- Hofstadter's Law: It always takes longer than you expect, even when you take into account Hofstadter's Law.

  - Douglas Hofstadter: Gödel, Escher, Bach: An eternal golden braid

Other agile approach: Kanban

- Visualize the workflow
- Limit WIP (stop starting & start finishing)
- Measure & optimize flow



Don’t hesitate to reach out!

Cristian Solomon
Project Manager
cristian.solomon@thoughtworks.com

thoughtworks

Bonus Slides

/thoughtworks university

March 2022

/thoughtworks

What is Thoughtworks university?

Thoughtworks university is a 3 week programme that introduces our newly hired graduates and career changers to the principles, practices and skills required to build working software.

Began in
2005
Bangalore

Previous office locations:
Pune, Xi'an &
Porto Alegre

Now Running
Remotely
With 3 regional groups
APAC, EUROPE, NA/LA

6
Global Terms
annually

49%
W&UGM attendees
in 2020

Over
3,571
attendees to date

TWU Learning Objectives

TWU is a key element within graduate hires' First Year Onboarding, focused on developing:

Customer Service Mindset & Skills

Technical and Consulting Skills

Culture / Way of Life at thoughtworks

Diversity and Inclusion

Programme Outline

Pre-course: TW101

Virtual learning modules covering general learning as well as role-based learning objectives, made up of reading material (books, articles), videos, e-learning modules and assignments (which are reviewed by the coaches).

3 week thoughtworks university term (currently running remotely)

- Client Engagement simulation
- Role specific dojos
- 1:1 coaching and feedback
- Remote networking events

These are the Primary Capabilities focused on in the curriculum:

- Agile Delivery Processes
- Architecture
- Build Quality In
- Build Security In
- Consulting
- Continuous Delivery
- Impact
- Software Development
- Cultivation
- Thoughtworks Culture & Way of Life
- Thoughtworks Values & Lenses

Thank you!

/thoughtworks