Project Planning

17-313 Fall 2023

Foundations of Software Engineering

https://cmu-313.github.io

Andrew Begel and Rohan Padhye

Learning Goals

- Recognize the importance of project planning
- Understand the difficulty of measuring progress
- Identify why software development has project characteristics
- Use milestones for planning and progress measurement
- Understand backlogs and user stories
- Get to know your team!

Project Teams

	Team 1	Team 2	Team 3	Team 4
	Lauren Smith	Chris Lee	Chayut Glankwamdee	Riya Bhatia
	Helen(Yujia) Zheng	Akintayo Salu	Aditya Ganesh	Jason Kwok
	Adeline Wu	Peter Khomchenko	Forever Akpabio	Mohamad El Ghali
	Yuvanshu Agarwal	Jaden D'Abreo	Keerti Mukkamala	Yerim Song
	Sherry Zhuge	Ryan Wong		
Section A				
	Team 1	Team 2	Team 3	Team 4
	Olivia Van Zandt	Yuhe Ma	Serena Yao	Sen Feng
	Hanah Ryu	Mayar Alkurdi	Alice Hong	Ariel Kwak
	Ke Hao Chen	Matthew Leboffe	Yuchen Liang	Cindy Liu
	Raunak Sood	Ritu Pathak	Sicheng Lu	Hao Kang
	Mimi Chuang		Bo Xia	Bojun Li
Section B				
	Team 1	Team 2	Team 3	Team 4
	Andrew Chung	Derek Kim	Janie Xiong	John Bakhtiyorjon Mirzajonov
	Julia Liu	Monica Qiu	Anastasiia Runova	Muhammad Ammar Raza
	Xuchao Zhou	Phyllis Feng	Sophia King	Emma Shi
	Aaron Marmolejos	Swati Anshu	Tanner Balluff	Kelly Cha
Section C		Luna Wei	Alexis Duong	Kaitlyn Liu

Project Teams

	Team 1	Team 2	Team 3	Team 4
	Neha Tirumalai	Reva Poddar	Tika Kumar	Sally Pak
	Ava Givone	Jacky Gao	Grace Liao	Lara Marinov
	Abby Chen	Zoe	Rashmi Francis	Adam Bournes
	Vania Halim	Holly Wang	Benjamin Chen	Meghna Chityala
Section D		Jonathan Lindstrom		Melody Wang
	Team 1	Team 2	Team 3	Team 4
	Simran Bedi	Minjoo Kim	Tze Hng Loke	Hank Xu
	Huarui Lai	Jonathan Ho	Kyle Chen	Rachel Luo
	Dhruva Reddy	Girase Nitya	Adrienne Li	Caleb Koo
	Aarav Tanti	Aanya Rustogi	Tanay Bennur	Sara Riyad
Section E				Tony Li
	Team 1	Team 2	Team 3	Team 4
	Constantine Westerink	Khuslen Misheel	Eichel Choi	Jordi Gonzalez
	Ellen Fang	Joyce Huang	Jamie Chen	Noor Mostafa
	Vanessa Lin	Tyrece Jeffrey	Oleg Plisov	Alysson Gu
	Sebastian Lu	Connor Maas	Kaylin Yeoh	Rachel Wu
Section F	Katelyn Zheng	Ryan Huang	Itamar Hindi	

Administrivia

- P1 due tonight 11:59pm
- We heard that some of your PRs don't pass all the tests.

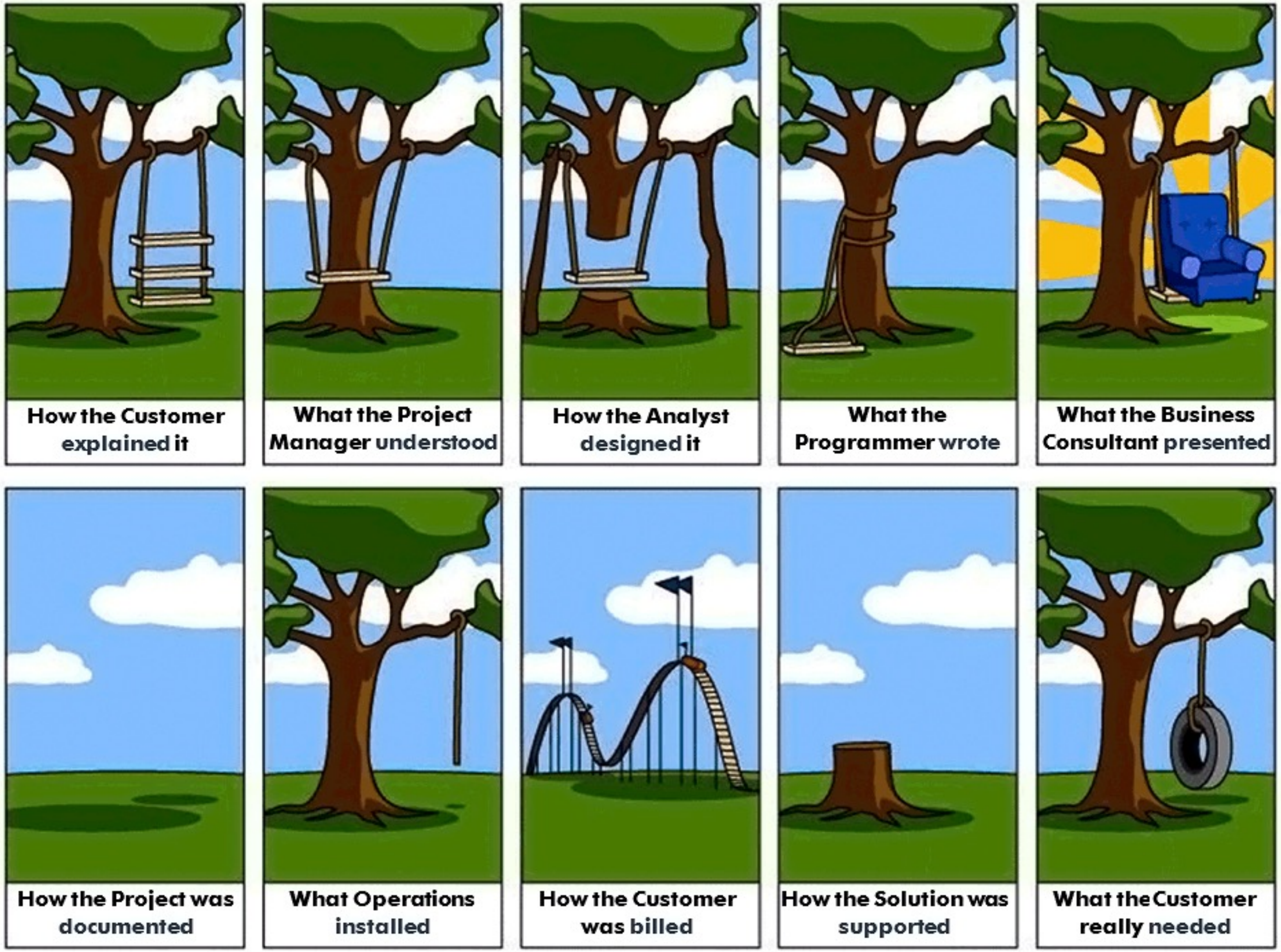
Administrivia

- Just write down what you did to try to debug it in your PR.
- See Prof. Padhye's Sept 6, 4:45pm post in #general on Slack.
- P2 will be released tomorrow (Friday Sept 7).
    - This is a group assignment.
    - Due date: October 12, 11:59pm.

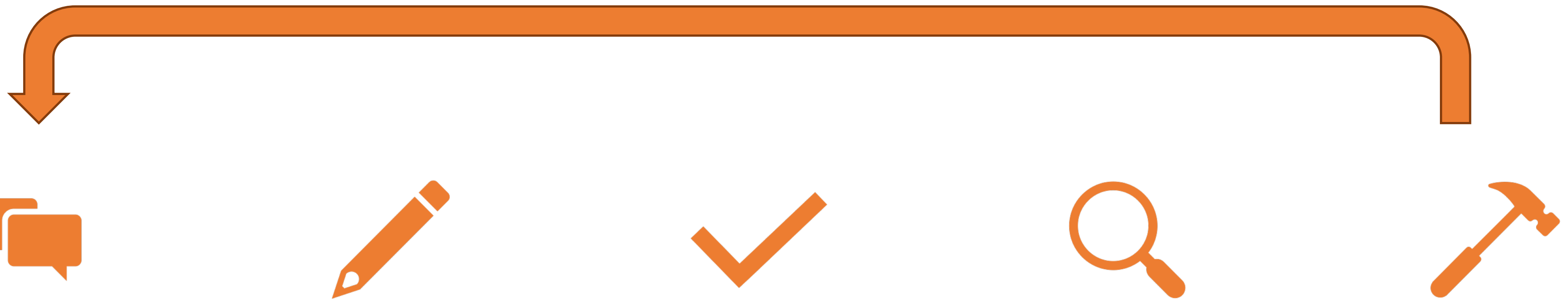
Software Process

"The set of activities and associated results that produce a software product."

Sommerville, *Software Engineering*, ed. 8


How the Customer explained it
What the Project Manager understood
How the Analyst designed it
What the Programmer wrote
What the Business Consultant presented
How the Project was documented
What Operations installed
How the Customer was billed
How the Solution was supported
What the Customer really needed

All software development process



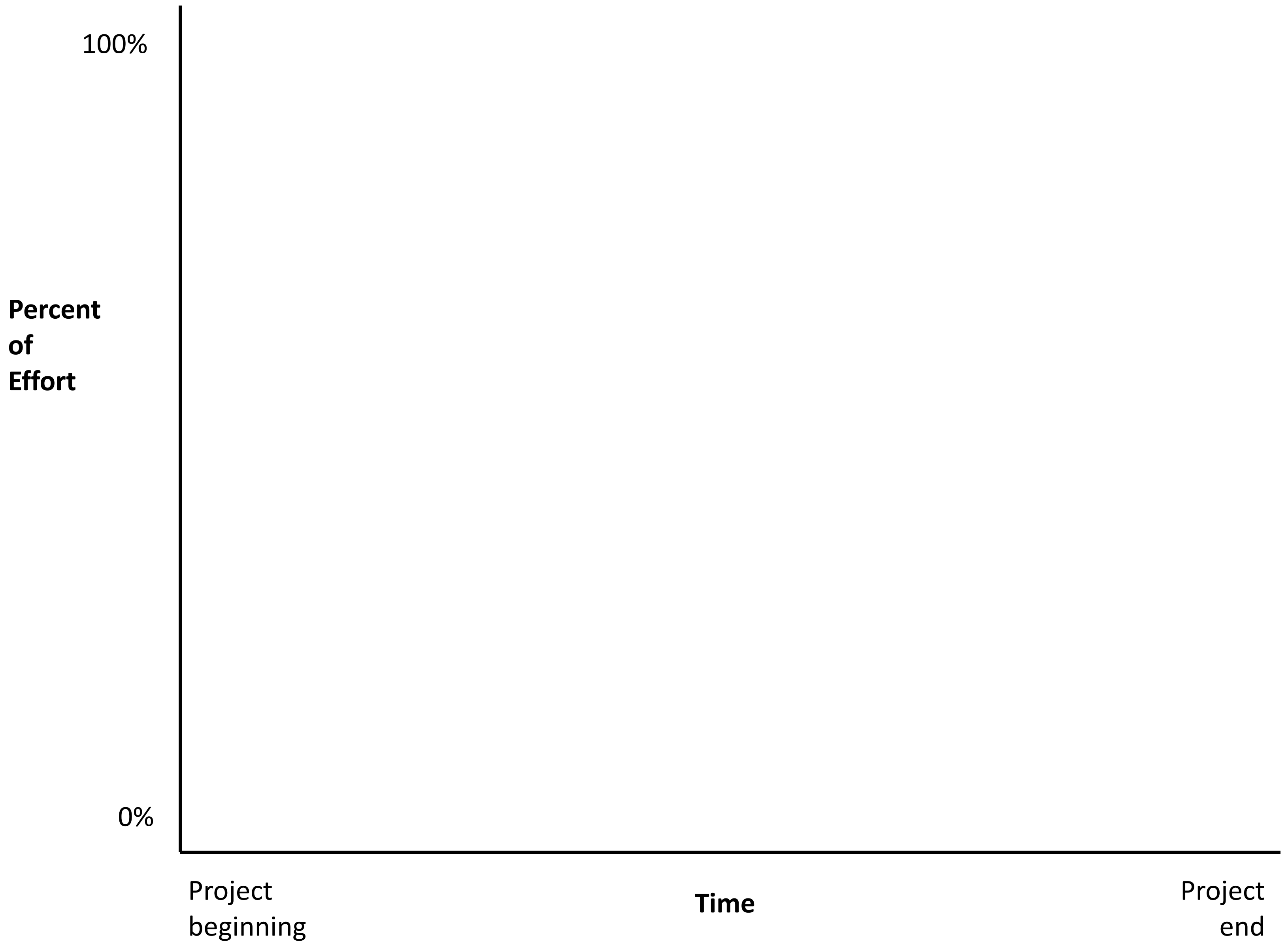
Discuss the software that needs to be written

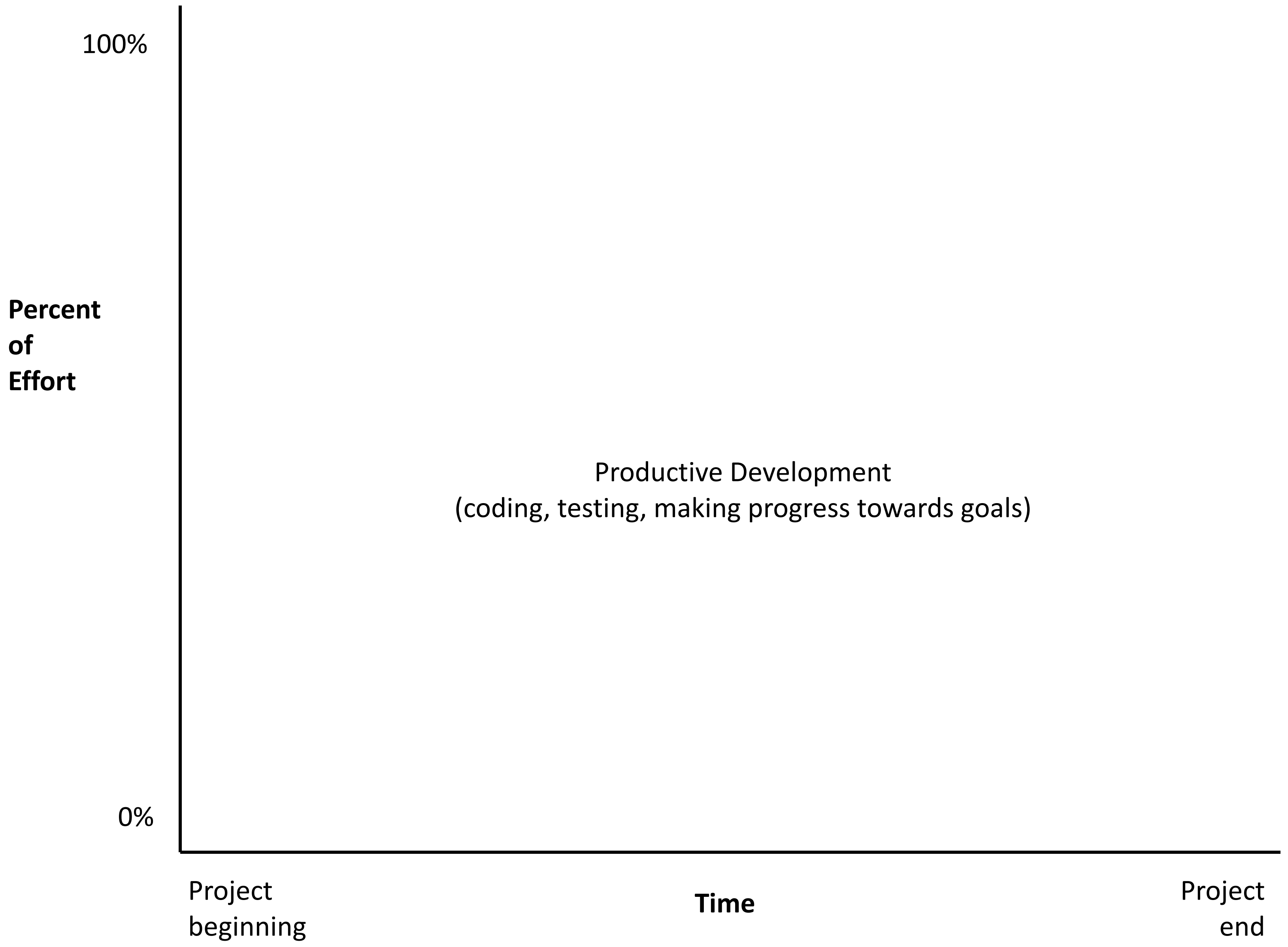
Write some code

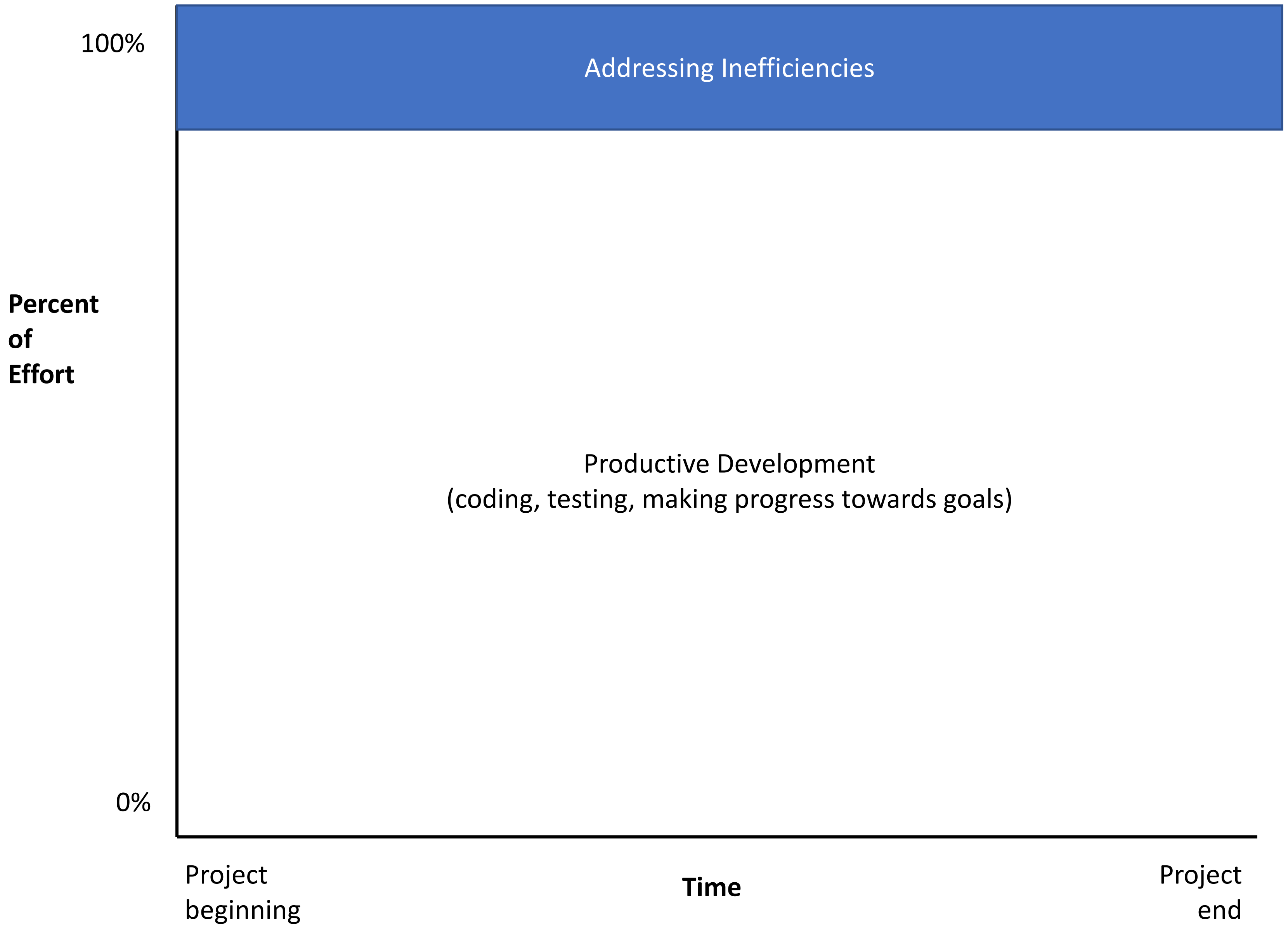
Test the code to identify the defects

Debug to find causes of defects

Fix the defects

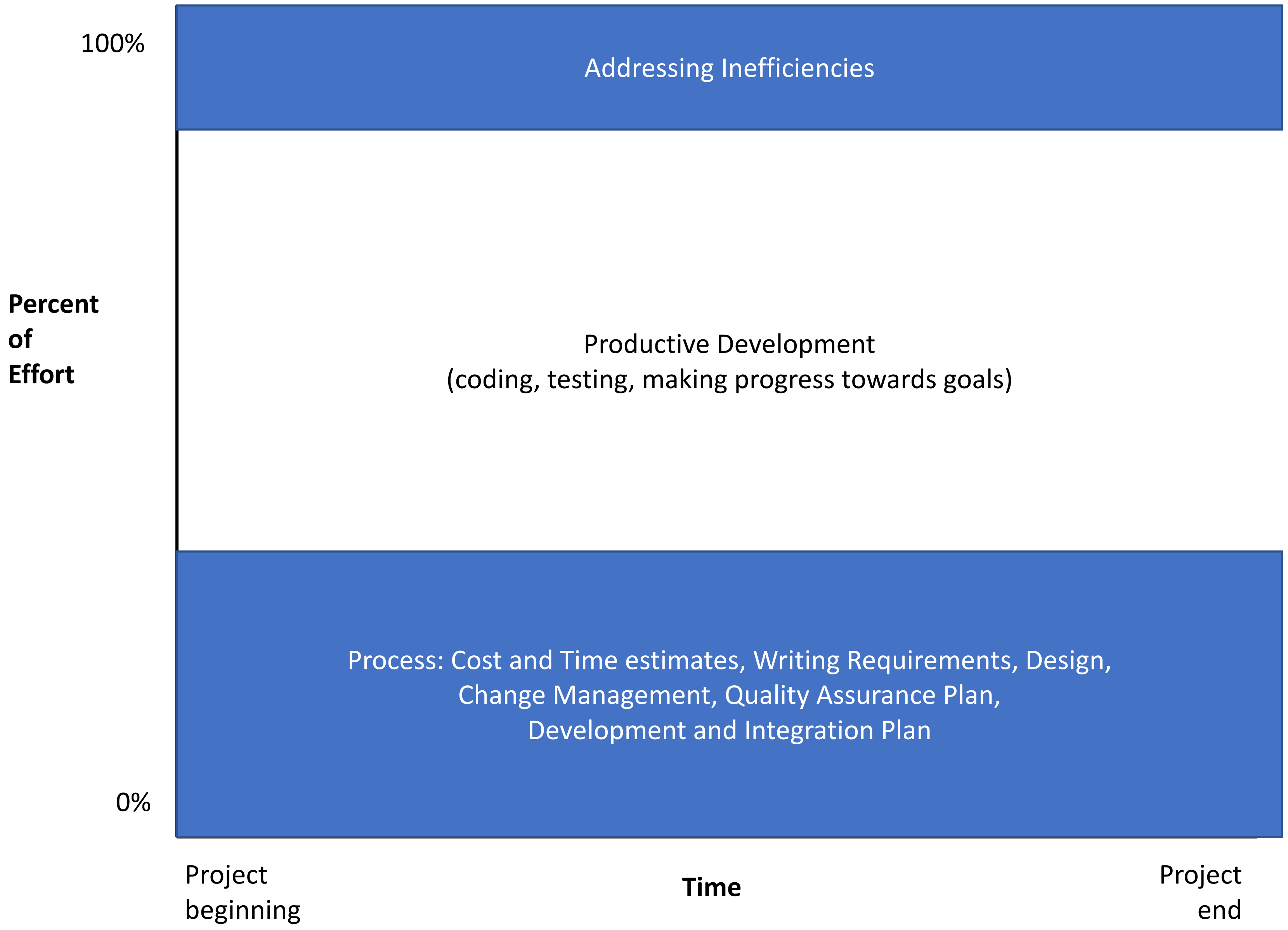

100%
Percent of Effort
0%
Project beginning
Time
Project end

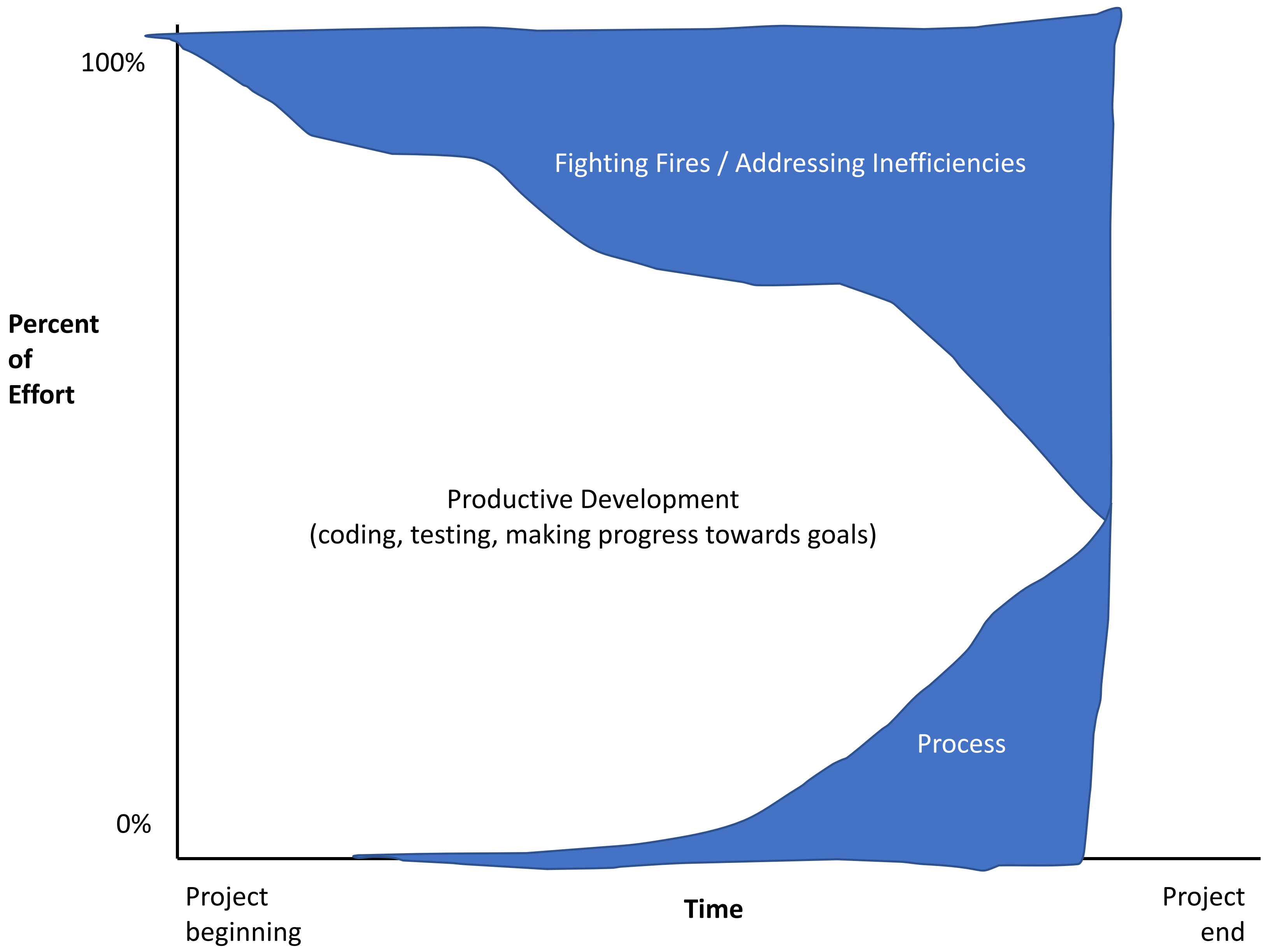

100%
Percent of Effort
Productive Development
(coding, testing, making progress towards goals)
0%
Project beginning
Time
Project end


100%
Addressing Inefficiencies
Percent of Effort
Productive Development
(coding, testing, making progress towards goals)
0%
Project beginning
Time
Project end

Let’s improve the reliability of this process

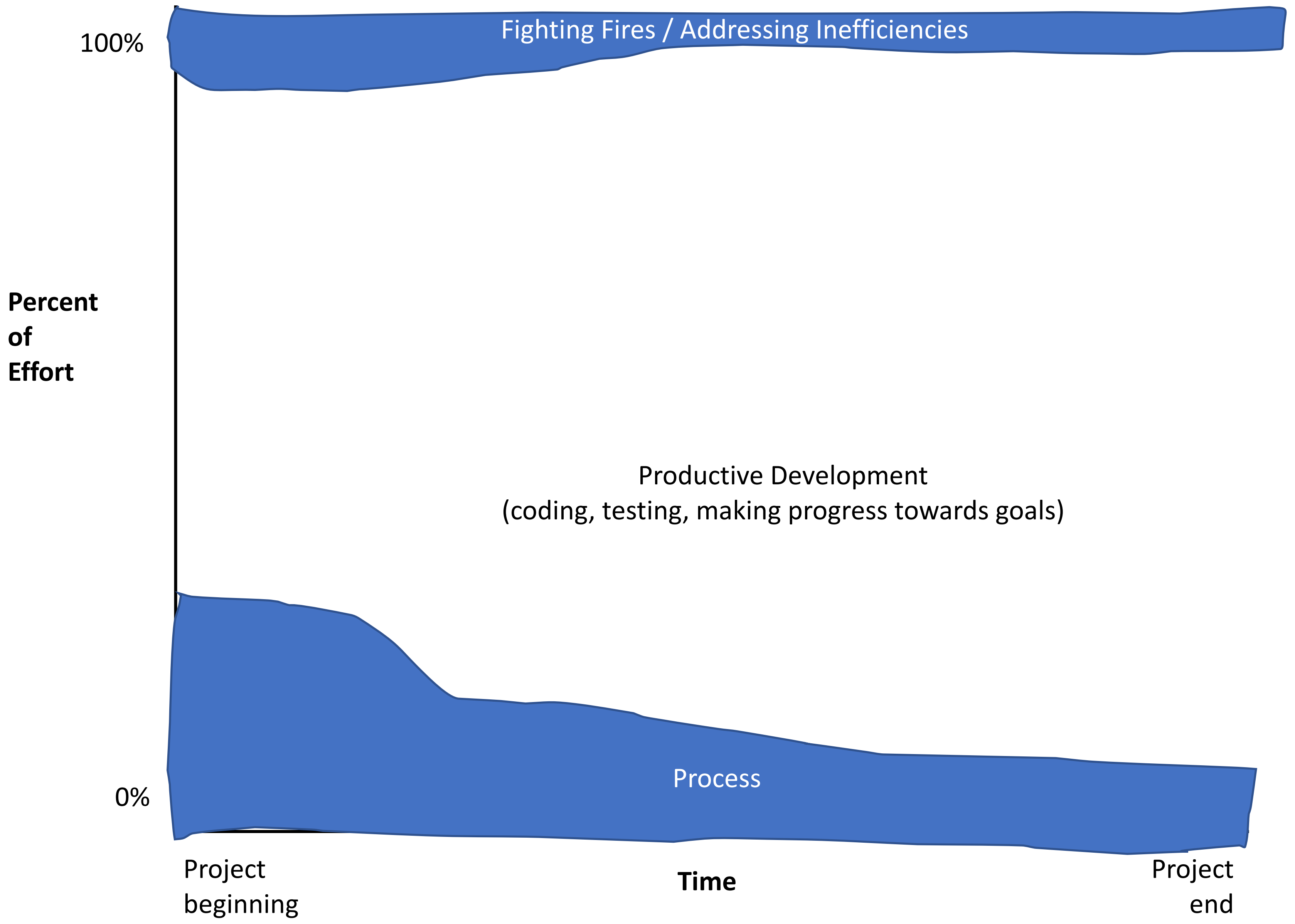
- Write down all requirements
  - Review requirements
  - Require approval for all changes to requirements
- Use version control for all changes
  - Review code
- Track all work items
  - Break down feature development into small tasks
  - Write down and monitor all reported bugs
  - Hold regular, frequent status meetings
- Plan and conduct quality assurance
- Employ a DevOps framework to push code between developers and operations


100%
Addressing Inefficiencies
Percent of Effort
Productive Development
(coding, testing, making progress towards goals)
Process: Cost and Time estimates, Writing Requirements, Design, Change Management, Quality Assurance Plan, Development and Integration Plan
0%
Project beginning
Time
Project end


100%
Percent
of
Effort
Fighting Fires / Addressing Inefficiencies
Productive Development
(coding, testing, making progress towards goals)
Process
0%
Project
beginning
Time
Project
end

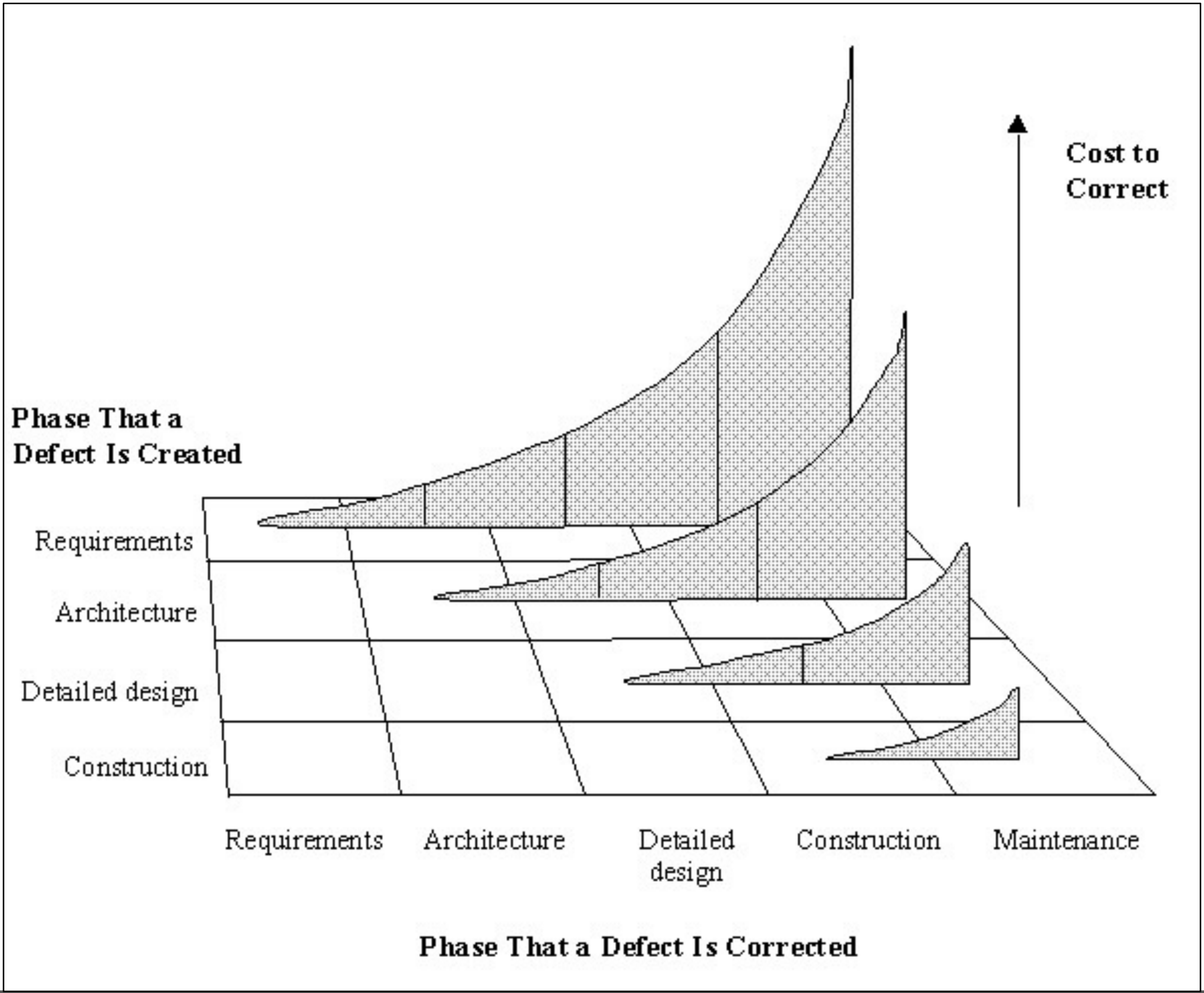
Example process issues

- Change Control: Mid-project informal agreement to changes suggested by customer. *Project scope expands 25-50%*
- Quality Assurance: Late detection of requirements and design issues. Test-debug-reimplement cycle limits development of new features. *Release with known defects.*
- Defect Tracking: Bug reports collected informally. *Bugs are overlooked.*
- System Integration: Integration of independently developed components at the very end of the project. *Interfaces out of sync.*
- Source Code Control: Accidentally overwrote changes. *Lost work.*
- Scheduling: Late project. *Developers asked to re-estimate work effort weekly.*



Hypothesis: Process increases flexibility and efficiency

Ideal Curve: Upfront investment for later greater returns


Cost to Correct
Phase That a Defect Is Created
Requirements
Architecture
Detailed design
Construction
Requirements
Architecture
Detailed design
Construction
Maintenance
Phase That a Defect Is Corrected

Planning

Time estimation

I'M JUST OUTSIDE TOWN, SO I SHOULD BE THERE IN FIFTEEN MINUTES.
ACTUALLY, IT'S LOOKING MORE LIKE SIX DAYS.
NO, WAIT, THIRTY SECONDS.
THE AUTHOR OF THE WINDOWS FILE COPY DIALOG VISITS SOME FRIENDS.

https://xkcd.com/612/

Activity: Estimate Time

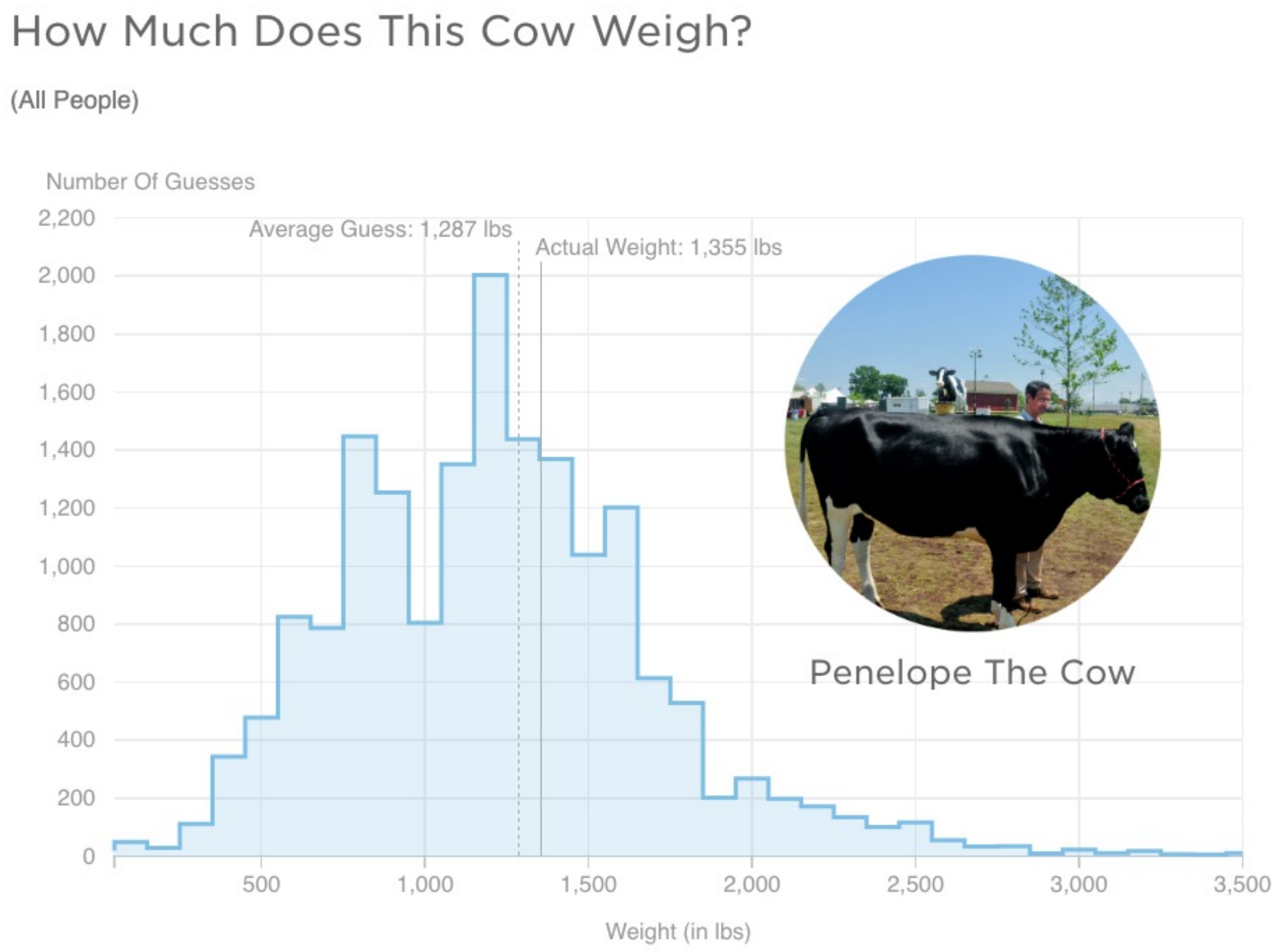
- Task A: Web version of the Monopoly board game with Pittsburgh street names
  - Team: just you
- Task B: Bank smartphone app
  - Team: you with team of 4 developers, one experienced with iPhone apps, one with background in security
- Estimate 8h days, 20 workdays in a month, 220 workdays per year

- My Task A estimate: ___ days/wks
  My Task B estimate: ___ days/wks
- Other Task A estimate: __ days/wks
  Other Task B estimate: __ days/wks
- Other Task A estimate: __ days/wks
  Other Task B estimate: __ days/wks

Revise Time Estimate

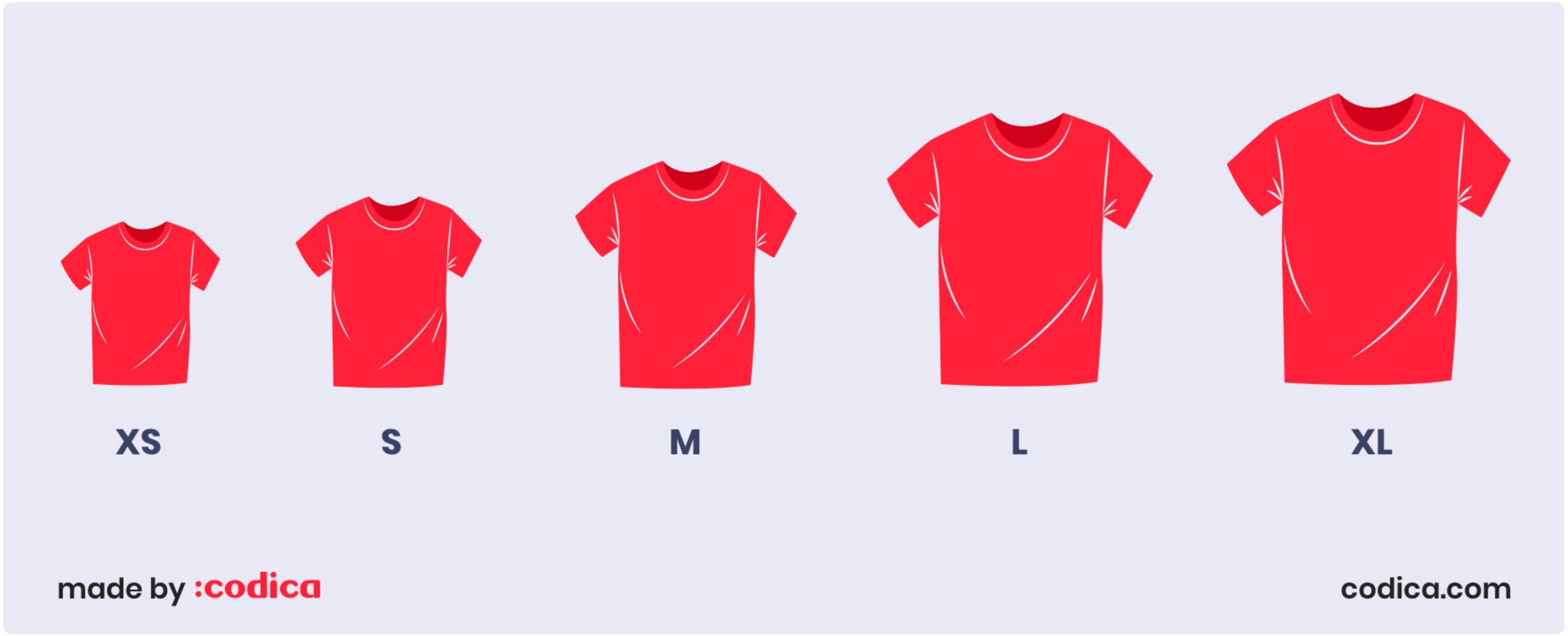
- Do you have comparable experience to base an estimate on?
- How much design do you need for each task?
- How much testing time do you need for each task?
- Let’s break down the task into ~5 smaller tasks and estimate their lengths.
- Revise our overall estimate, if necessary

Wisdom of the Crowd



Source: The Internet.

Credit: Quoctrung Bui/NPR


XS
S
M
L
XL
made by :codica
codica.com

$$\times\pi$$

Measuring Progress?

- "I'm almost done with the app. The frontend is almost fully implemented. The backend is fully finished except for the one stupid bug that keeps crashing the server. I only need to find the one stupid bug, but that can probably be done in an afternoon. We should be ready to release next week."

Measuring Progress?

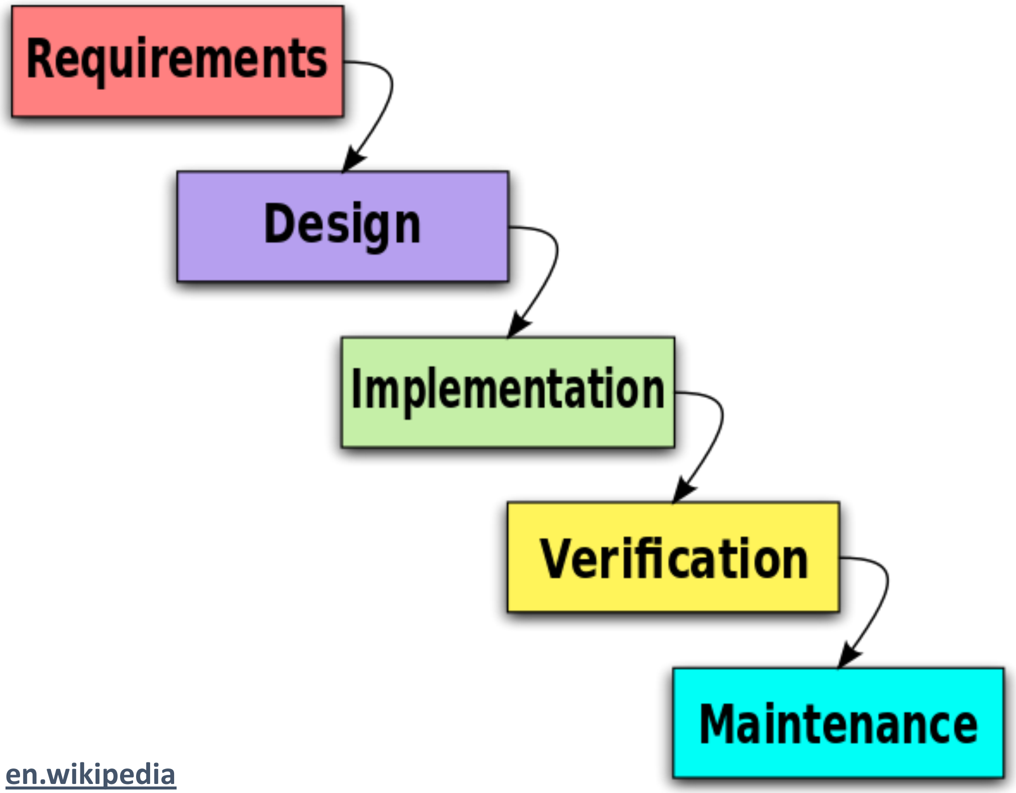
- Developer judgment: x% done
- Lines of code?
- Functionality?
- Quality?

Milestones and deliverables make progress observable

- Milestone: clear end point of a (sub)tasks
  - For project manager
  - Reports, prototypes, completed subprojects
  - “80% done“ is not a suitable mile stone
- Deliverable: Result for customer
  - Similar to a milestone, but for customers
  - Reports, prototypes, completed subsystems

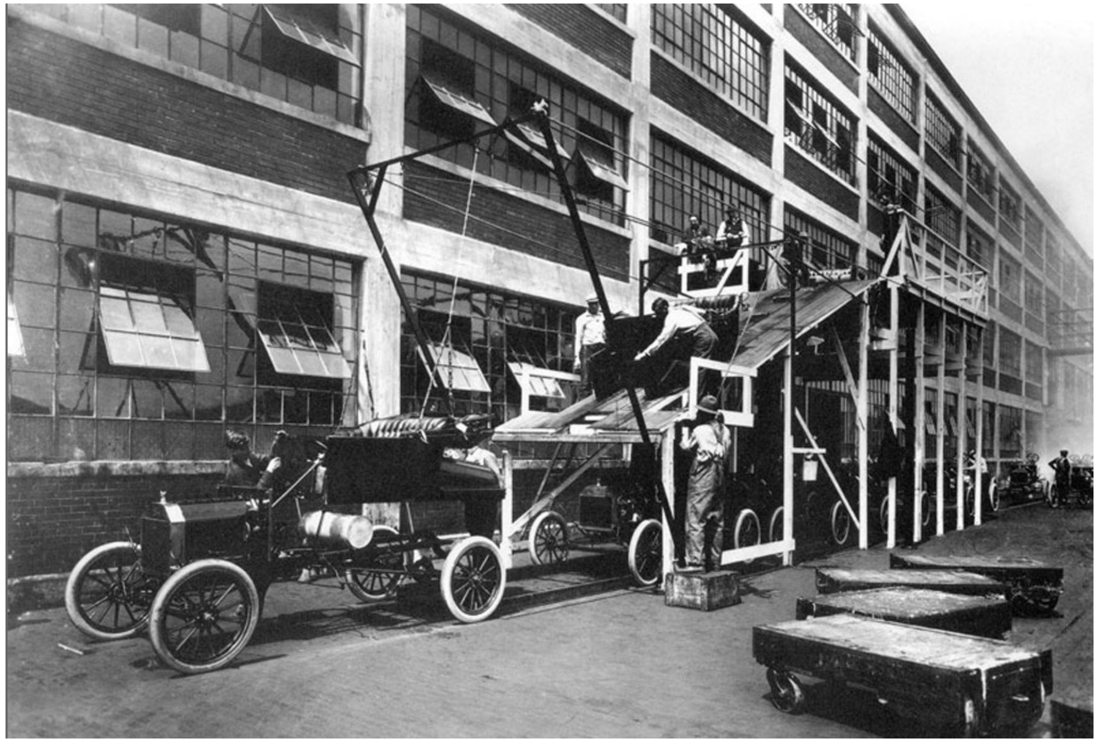
Processes

Waterfall was the OG software process



Waterfall diagram CC-BY 3.0 **Paulsmith99 at en.wikipedia**

… akin to processes pioneered in mass manufacturing (e.g., by Ford)



Lean production adapts to variable demand

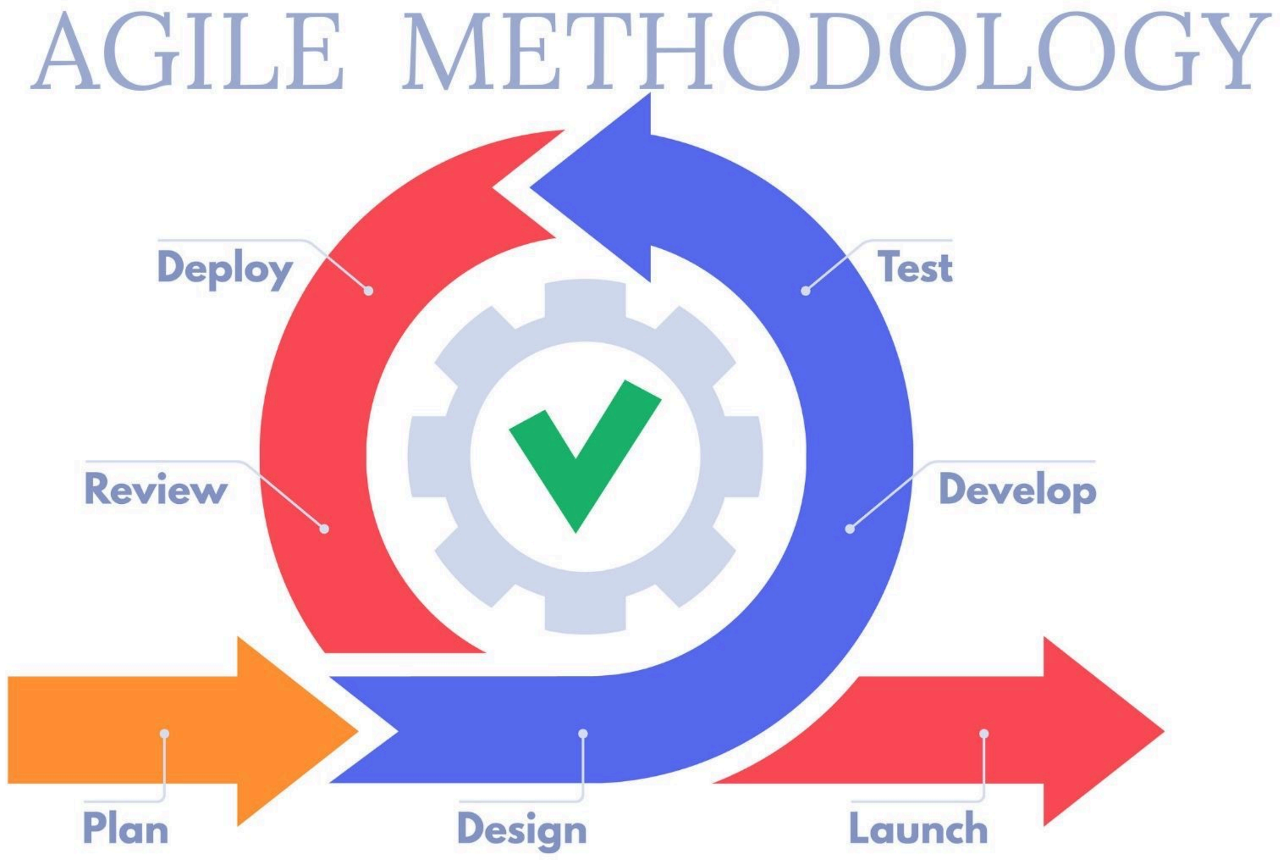
Taiichi Ohno

- Toyota Production System (TPS)
  - Build only what is needed, only when it is needed.
  - Use the "pull" system to avoid overproduction (Kanban)
  - Stop to fix problems, to get quality right from the start (Jidoka)
  - Workers are multi-skilled and understand the whole process; take ownership

- Lots of recent software buzzwords build on these ideas
  - Just-in-time, DevOps, Shift-Left

See also: "The machine that changed the world" by James P Womack et al. The Free Press, 2007.

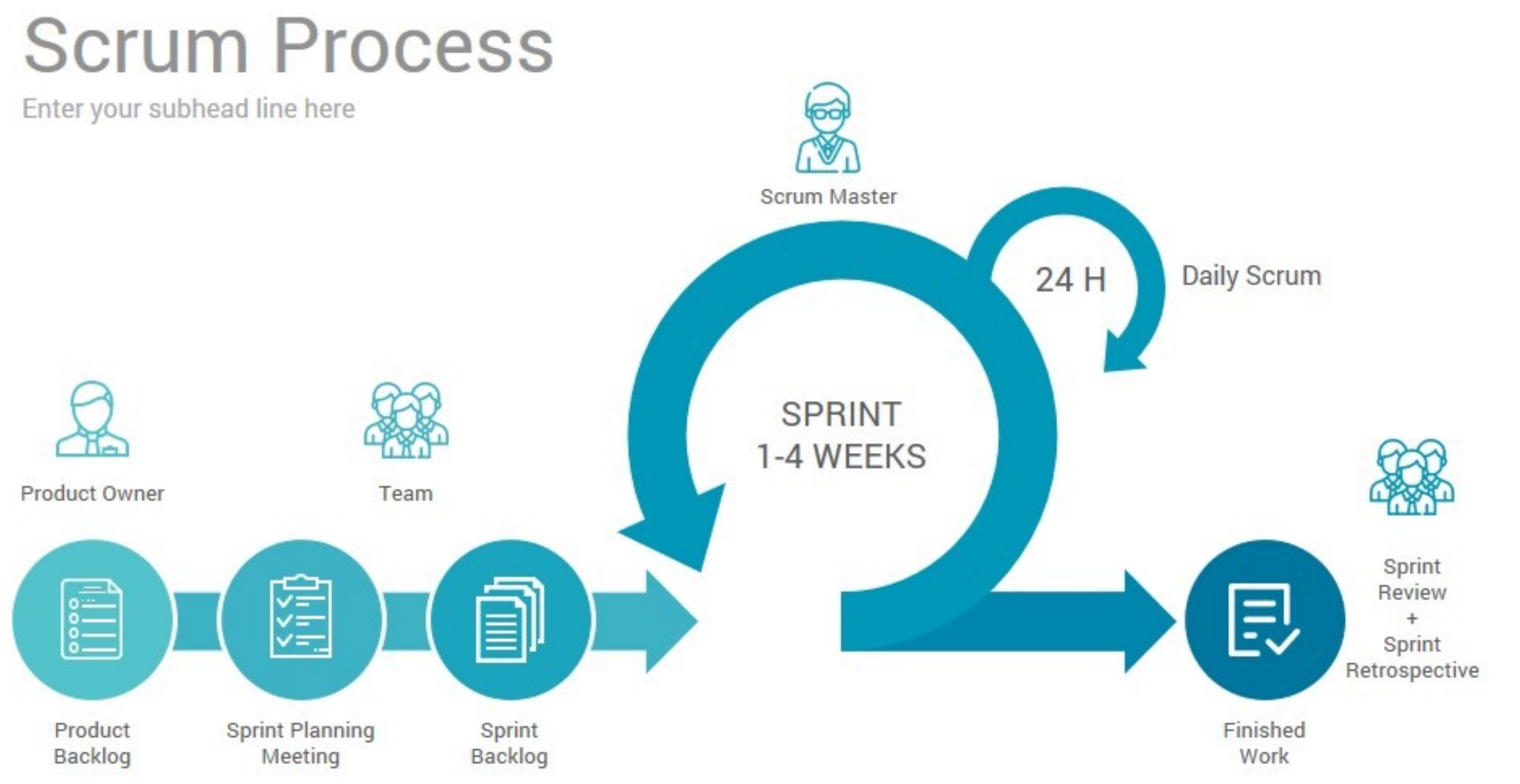
Now, most of us use Agile Methods



Scrum

(Only a brief intro)

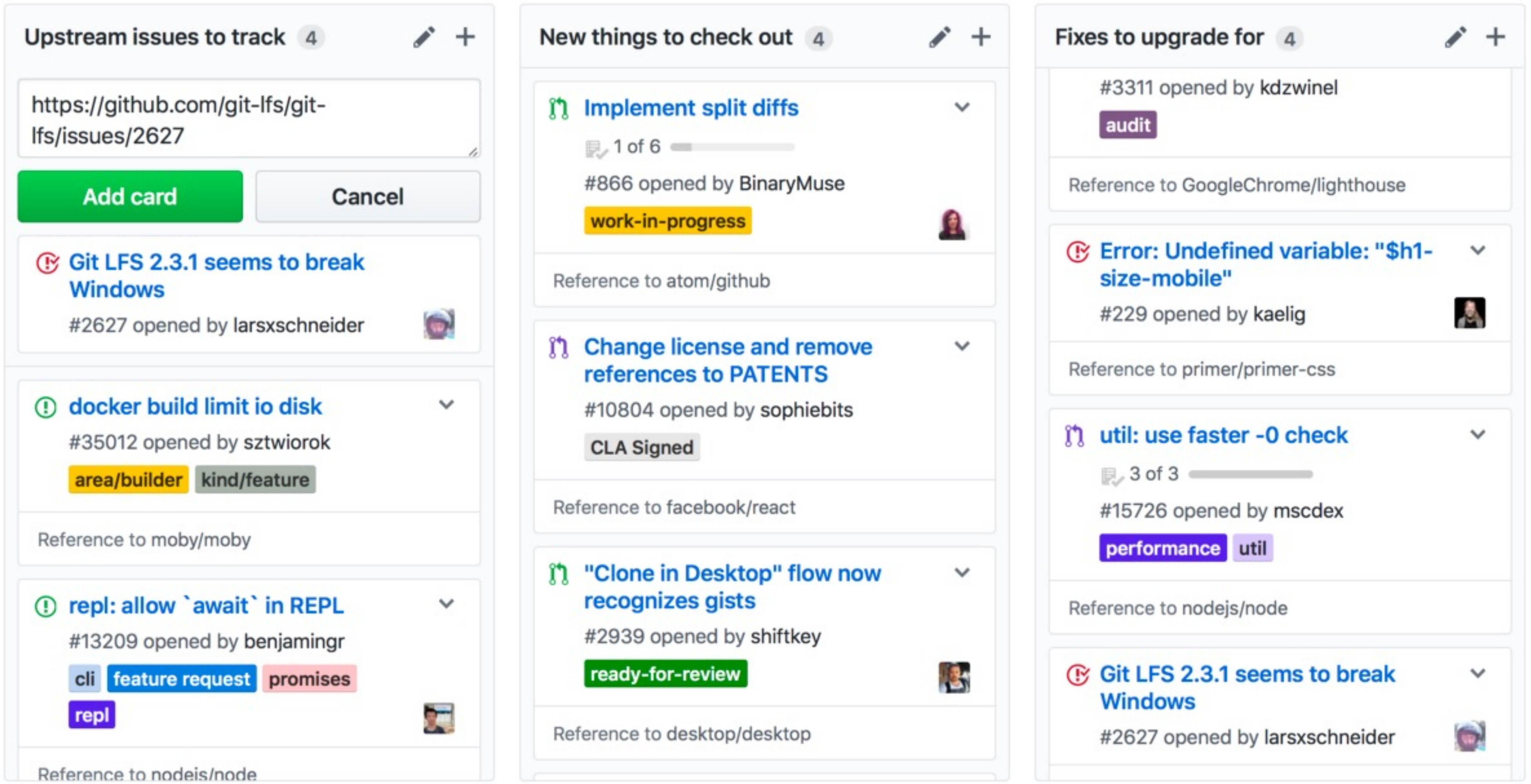
Elements of Scrum



Backlogs

- The product backlog is all the features for the product
- The sprint backlog is all the features that will be worked on for that sprint. These should be broken down into discrete tasks:
    - Fine-grained
    - Estimated
    - Assigned to individual team members
    - Acceptance criteria should be defined
- User Stories are often used

Kanban boards



Upstream issues to track 4

https://github.com/git-lfs/git-lfs/issues/2627

Add card | Cancel

Git LFS 2.3.1 seems to break Windows
#2627 opened by larsxschneider

docker build limit io disk
#35012 opened by sztwiorok
area/builder kind/feature
Reference to moby/moby

repl: allow `await` in REPL
#13209 opened by benjamingr
cli feature request promises repl
Reference to nodejs/node

New things to check out 4

Implement split diffs
1 of 6
#866 opened by BinaryMuse
work-in-progress
Reference to atom/github

Change license and remove references to PATENTS
#10804 opened by sophiebits
CLA Signed
Reference to facebook/react

"Clone in Desktop" flow now recognizes gists
#2939 opened by shiftkey
ready-for-review
Reference to desktop/desktop

Fixes to upgrade for 4

#3311 opened by kdzwinel
audit
Reference to GoogleChrome/lighthouse

Error: Undefined variable: "$h1-size-mobile"
#229 opened by kaelig
Reference to primer/primer-css

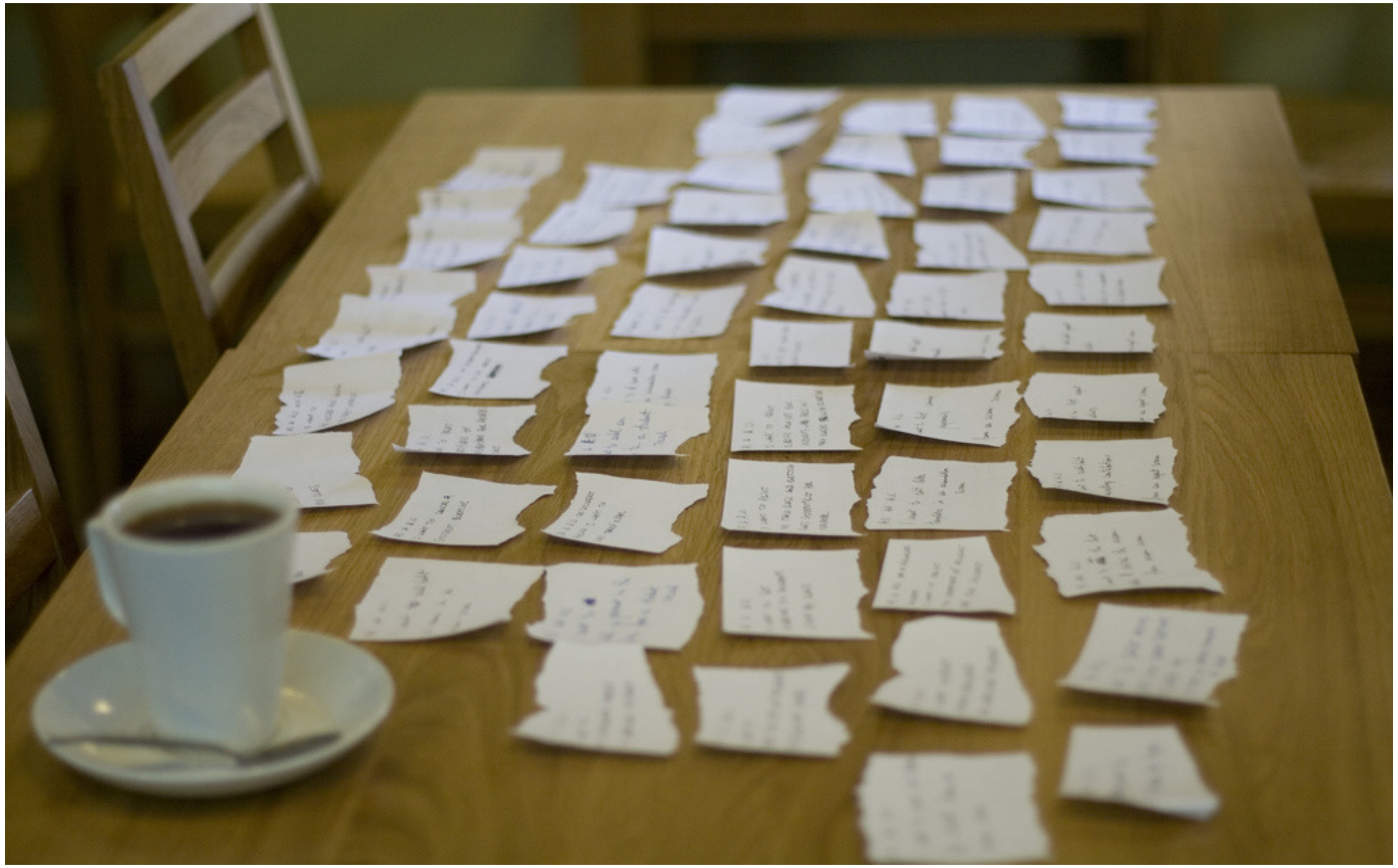
util: use faster -0 check
3 of 3
#15726 opened by mscdex
performance util
Reference to nodejs/node

Git LFS 2.3.1 seems to break Windows
#2627 opened by larsxschneider

Scrum Meetings

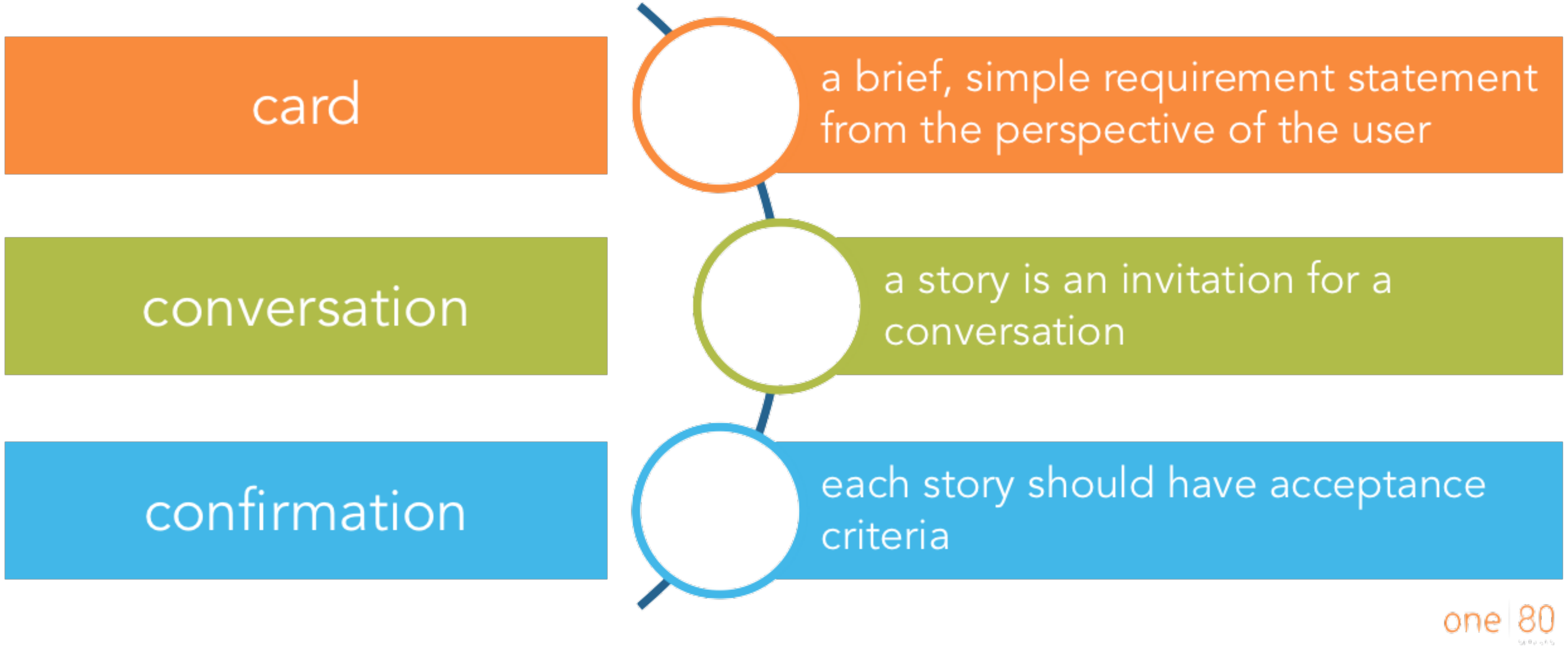
- Sprint Planning Meeting
  - Entire Team decides together what to tackle for that sprint
- Daily Scrum Meeting
  - Quick Meeting to touch base on :
    - What have I done? What am I doing next? What am I stuck on/need help?
- Sprint Retrospective
  - Review sprint process
- Sprint Review Meeting
  - Review Product

User Stories



Source: https://www.flickr.com/photos/jakuza/2728096478

User Stories



card — a brief, simple requirement statement from the perspective of the user

conversation — a story is an invitation for a conversation

confirmation — each story should have acceptance criteria

Source: http://one80services.com/user-stories/writing-good-user-stories-hint-its-not-about-writing/

Card

- “As a [role], I want [function], so that [value]”

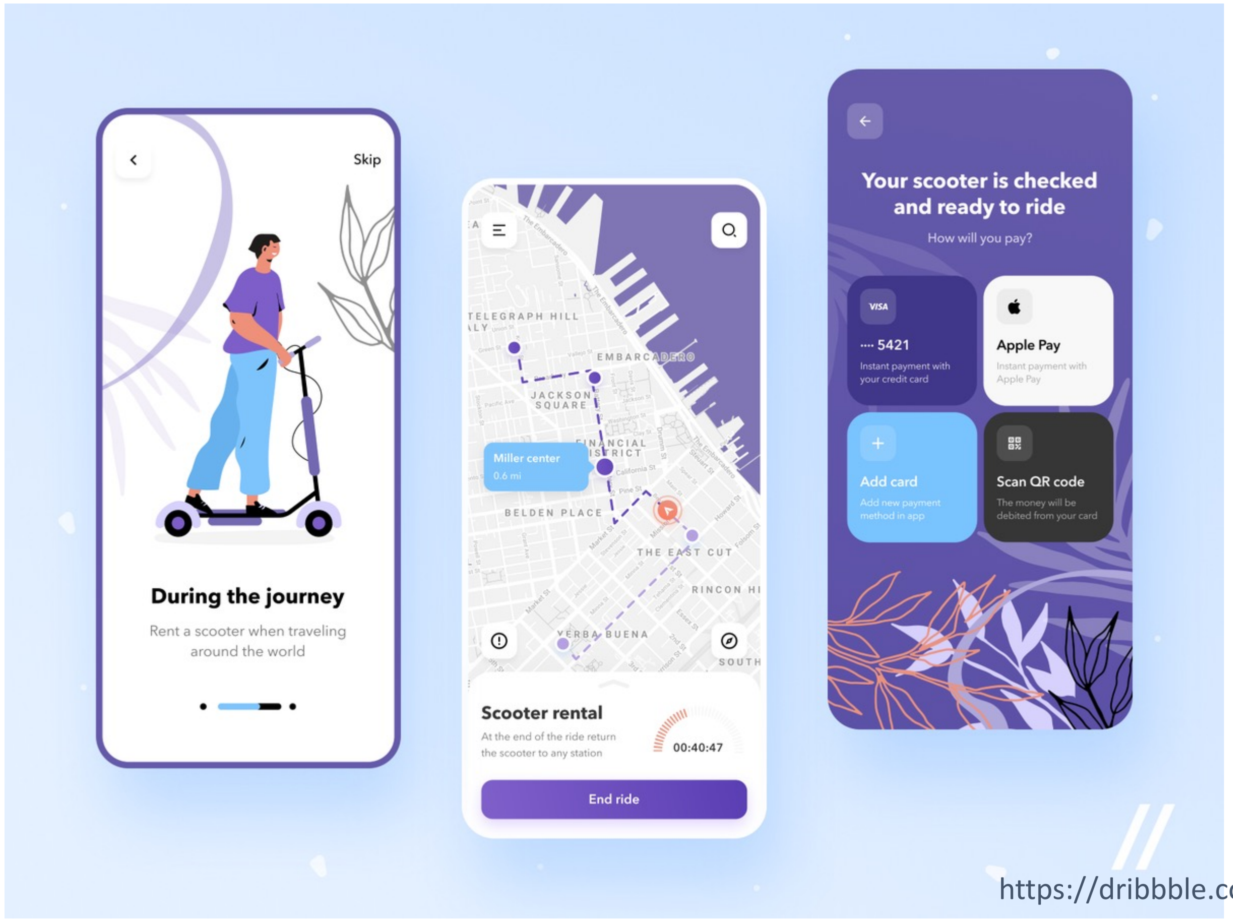
Conversation

- What must a developer do to implement this user story?

Confirmation

- How can we tell that the user story has been achieved?
- It’s easy to tell when the developer finished the code.
- But, how do you tell that the customer is happy?

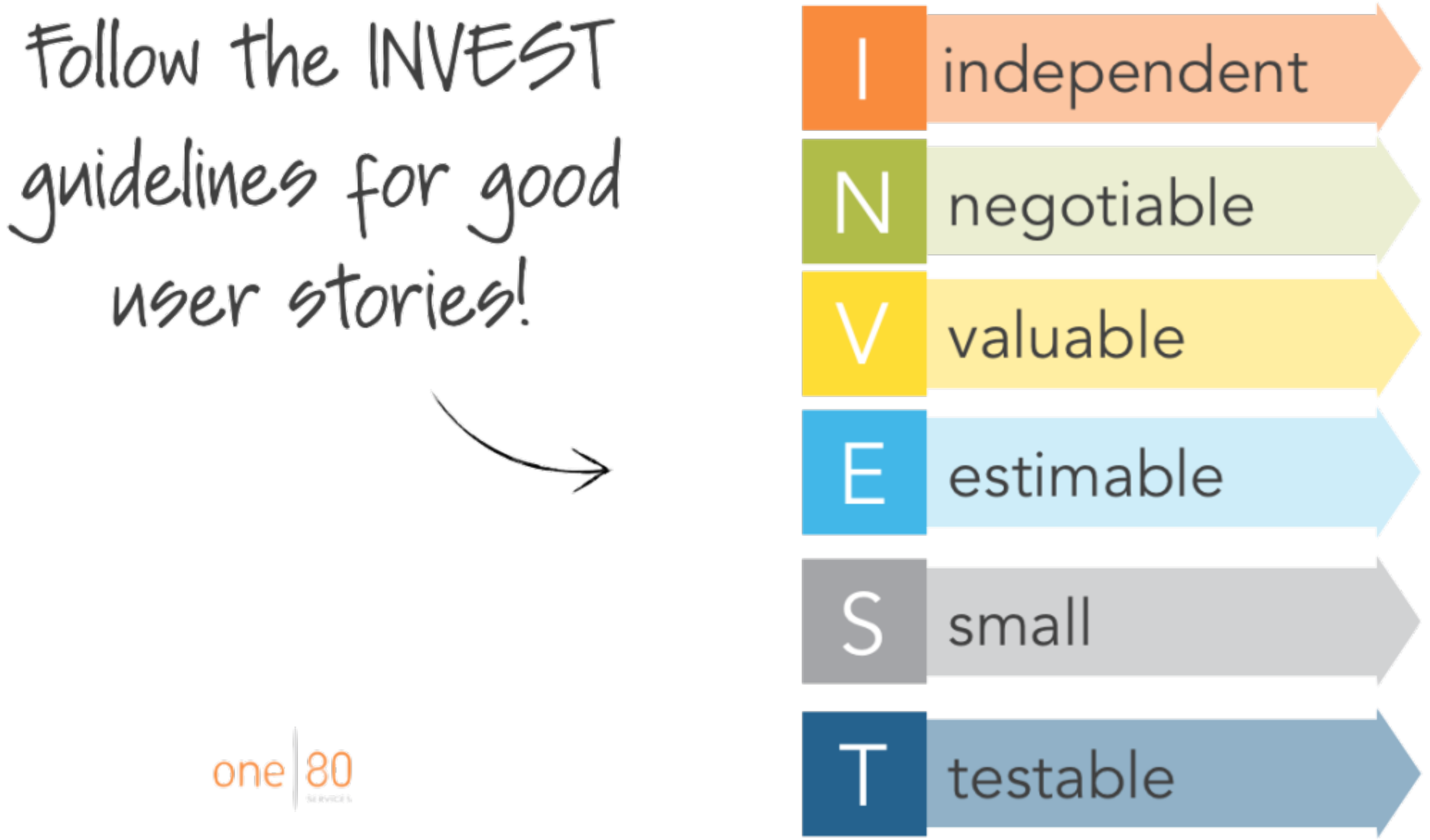
Exercise



Skip
During the journey
Rent a scooter when traveling around the world
Miller center
0.6 mi
TELEGRAPH HILL
EMBARCADERO
JACKSON SQUARE
FINANCIAL DISTRICT
BELDEN PLACE
THE EAST CUT
RINCON HI
YERBA BUENA
SOUTH
Scooter rental
At the end of the ride return the scooter to any station
00:40:47
End ride
Your scooter is checked and ready to ride
How will you pay?
VISA
···· 5421
Instant payment with your credit card
Apple Pay
Instant payment with Apple Pay
Add card
Add new payment method in app
Scan QR code
The money will be debited from your card

https://dribbble.com/shots/12512417-Scooter-Rental-App-Design

How to evaluate a user story?



Source: http://one80services.com/user-stories/writing-good-user-stories-hint-its-not-about-writing/

I	independent
N	negotiable
V	valuable
E	estimable
S	small
T	testable

Independent

- Schedule in any order.
- Not overlapping in concept.
- Not always possible.

Negotiable

I independent
N negotiable
V valuable
E estimable
S small
T testable

- Details to be negotiated during development.
- A good story captures the essence, not the details.

Valuable

I independent
N negotiable
V valuable
E estimable
S small
T testable

- This story needs to have value to someone (hopefully the customer).
- Especially relevant to splitting up issues.

I	independent
N	negotiable
V	valuable
E	estimable
S	small
T	testable

Estimable

- Helps keep the size small.
- Ensure we negotiated correctly.
- “Plans are nothing, planning is everything”
  - Dwight D. Eisenhower

I	independent
N	negotiable
V	valuable
E	estimable
S	small
T	testable

Small

- Can be written on a 3x5 card.
- At most two person-weeks of work.
- Too big === unable to estimate

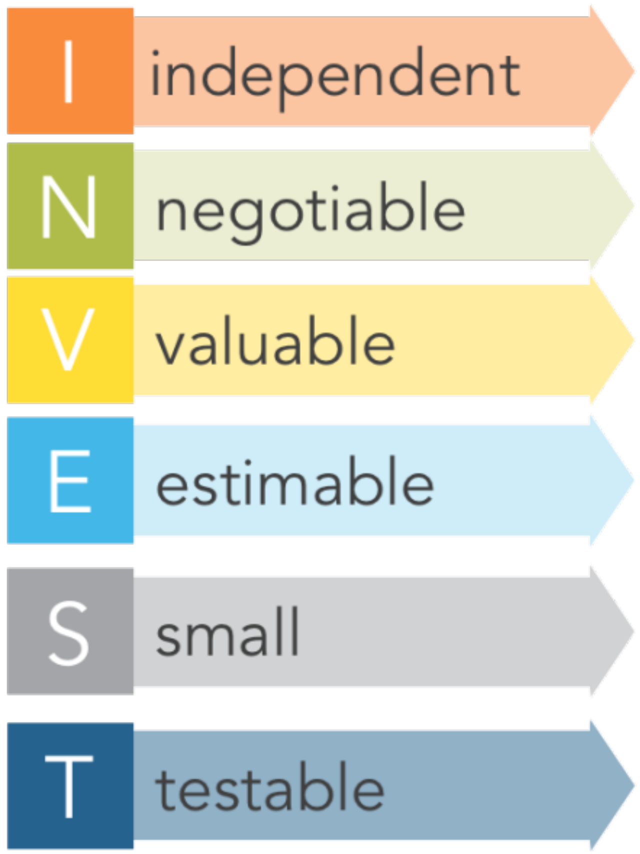
Testable

I independent
N negotiable
V valuable
E estimable
S small
T testable

- Ensures understanding of task
- We know when we can mark task “Done”
- Unable to test === I do not understand it

Activity

Follow the INVEST guidelines for good user stories!



I	independent
N	negotiable
V	valuable
E	estimable
S	small
T	testable

one|80

Next up: Teams