Lean Software Development

Week 1

What do we mean by Lean?

Lean is just like Agile

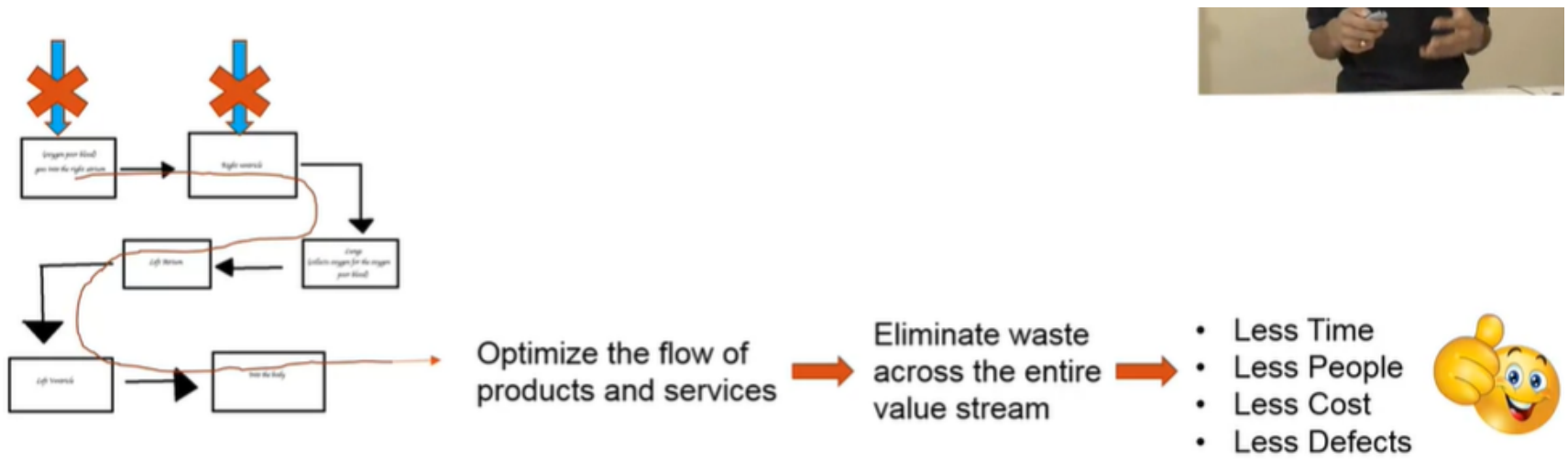
A set of principles, idea, lots and lots of tools/techinque

Origin: manufactoring - Toyota product development system

Core idea: focus on customer

- maximize value for your customer
- try to minimize waste in your process
- so that results in creating more value for the customer with minimum resources

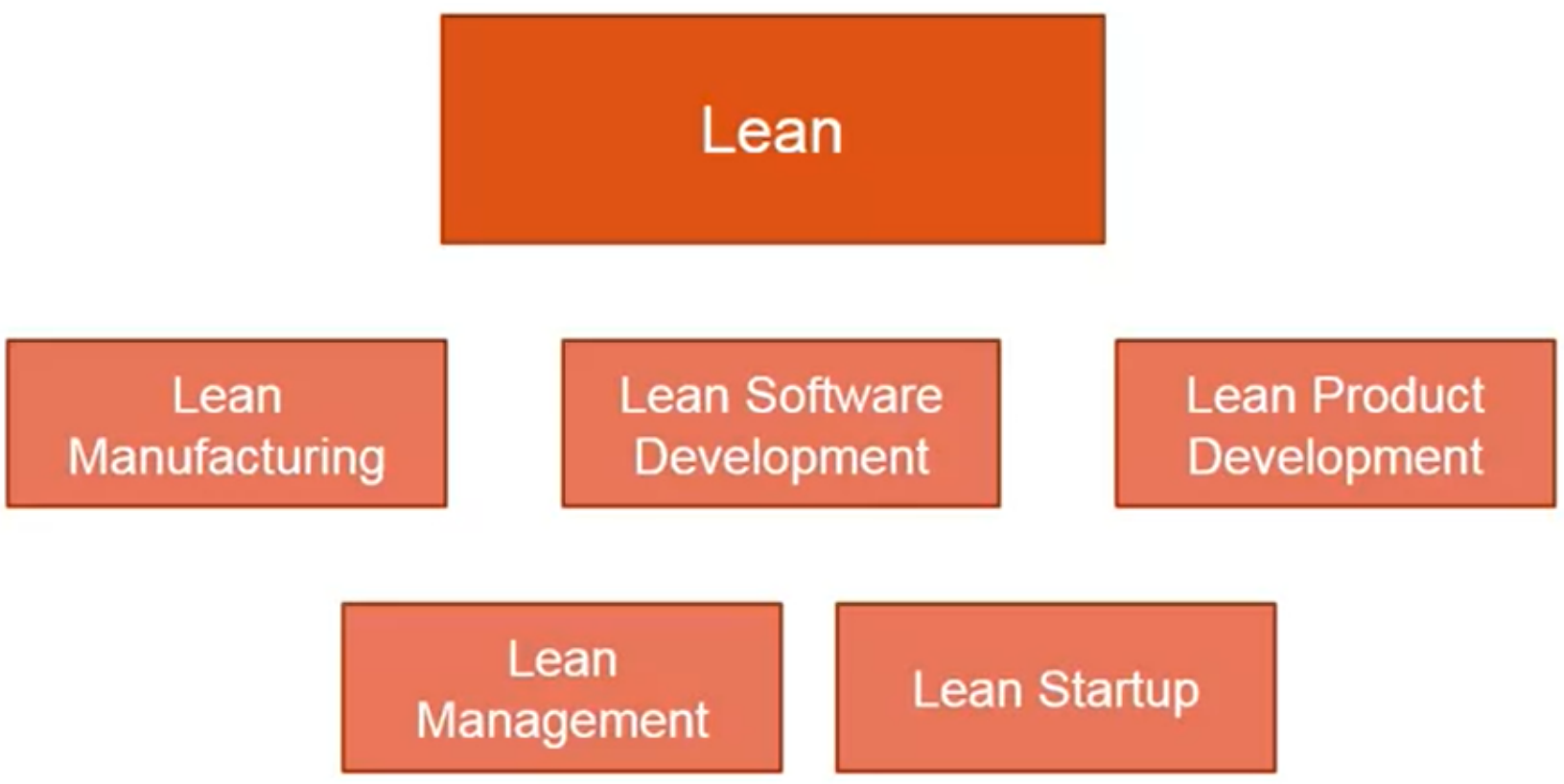
Instead of focus on optimizing the individual steps of process, try to focus on the flow of the process (optimize the flow of the product/services)



Is it only applicatble to products and manufactoring?

It is a way of thinking

Equality apply to services and other areas



Five Principles of Lean Manufacturing

- Identify Value

- Map the Value Stream
- Create Flow
- Establish Pull
- Seek Perfection

Value (from the customer’s perspective)

Value Stream (map the steps in the value stream)

Flow (create a smooth flow)

Pull (the right amount pulled at the right time, no more, no less)

Perfection (elimination of all waste in the value stream)

Lean for software development

The seven principles of lean software development

1. Eliminate waste
2. Amplify learning → Create knowledge
3. Defer commitment → Decide as late as possible
4. Deliver fast → Deliver as fast as possible
5. Empower the team → respect people
6. Built quality in → Built integrity in
7. Optimize the whole → see the whole

Kanban

Basically forces or encourages the team to optimize the whole flow

Lean startup

Not talk about can the solution be built, but it is about should the solution be built

Principles:

- Validated learning
  Try to validate your assumption through actual data
- Entrepreneurs are everywhere
- Entrepreneurship is management
- Innovative accounting

Lots of other lean techniques/ideas

- Value stream mapping
- Kaizen
- Theory of Constraints
- Cost Of Delay

Eliminate Waste

- Defects
- Overproduction
- Waiting
- Non-Standard Processing
- Transportation
- Intellect

- Motion
- Excess inventory

Missing or incorrect information (documentation) → **Defect**

Underutilizing team members by not matching their skillsets or capabilities with task/work needed it → **Intellect**

Excess bending and turning when repeatedly using materials/equiment for the job → **Motion**
(solution: relocating materials to a point of use)

Storing materials or documentation before they are needed → Excess inventory

Amplify Learning / Create Knowledge

What is it?

“To increase the ability of a team to learn quickly and effectively”

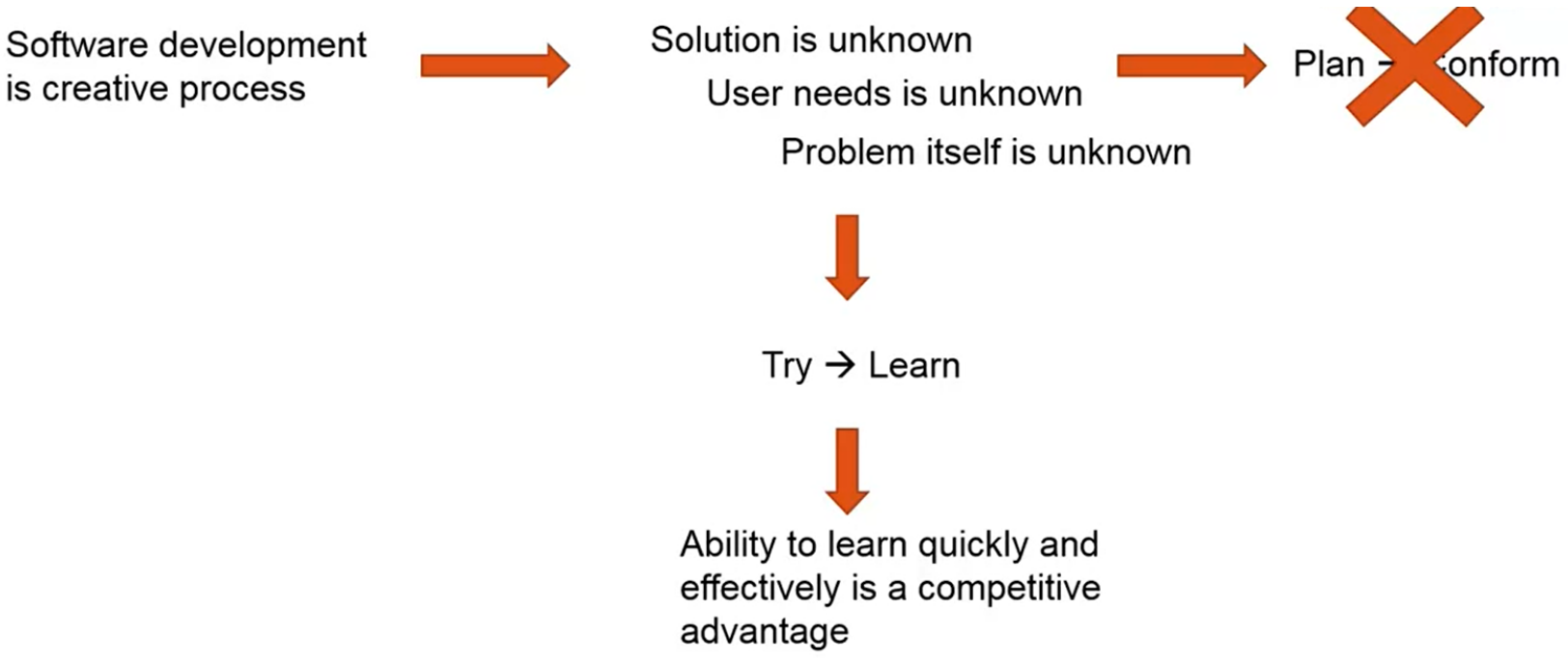
User needs → Solution → Process

Why does it matter?

Software development is creative process

→ Solution is unknown
User needs is unknown
Problem itself is unknown

→ ~~Plan - Confirm~~



How do you amplify learning?

Iterations - Short development cycles

Synchronization

- Spanning application: limited scope implemented end to end by advance team
- Develop interfaces together → Develop components individually
- Daily builds and smoke tests
  + Continuous code integration and automated tests

Set based development

- Development multiple options → Solution emerge

Defer commitments

Why decide when we know the least?

What is it?

“Ability to make commitment as late as possible. Make decision when you have the information you need.”

“does not mean delay the development”

Why does it matter?

- We have to make all sorts of decisions
- As project progresses, we have more information

→ The sooner we make a decision, the more chances of it being wrong → C stly

↓ Delay Decision until we have the information we need or it is really required

How do you defer commitment?

Concurrent vs. Sequential development

Ability to postpone decision until last responsible moment or make it possible to make changes easily so changed decisions can be digested effectively

Loosely coupled architecture

- Use modular design, Encapsulation, Separation of concern
- Use interfaces and abstraction
- Use parameters

Avoid repetition: change once and be done!

“Deliver fast” compliments this principle

Build Quality In

What is it?

“Don’t defer quality checks to a development phase. Instead, build quality form the start and at every step”

“Instead of tracking defects, try to prevent defects”

Why does it matter?

Defect → Waste

Easier to find & Easier to fix

Measure true progress

Defect masking

How do you build quality in?

Standards
+ Reduce conversion waste
+ Have to be used to be useful

Code reviews

Pairing

Mistake proofing → automation

Automated tests

Test driven development

Continuous integration

Awareness and Importance of Quality in team

Respect people

encouraging people to have their say. Having empathy for their point of view and trying to see things from their perspective.

giving people the responsibility to make decisions about their work

make sure that the empowered person still communicates about their intended approach, and why they think it’s the best approach.

discuss pros and cons and understand why the person wants to take that particular decision.

Deliver fast

What is it?

“Reduce development cycle time without compromising the quality”

Why does it matter?

Customer like faster delivery

Compliments “defer commiments” principle

Reduces risks / reduces waste

How do you deliver fast without compromising quality?

Reduce cycle time

- Little law: cycle time = things in process / Avg completion rate

Minimize number of items in process

Minimize size of things in progress

Limit work to capacity

Use pull scheduling

Optimize the whole

What is it?

“Consider the complete system when optimizing rather than individual component”

“Optimize the whole system rather than individual component”

Why does it matter?

Avoid optimzing the wrong component

In most cases, optimzing one component leads to surprising unintended consequences in other parts of system

A system is not just the sum of its parts - it is the product of their interactions

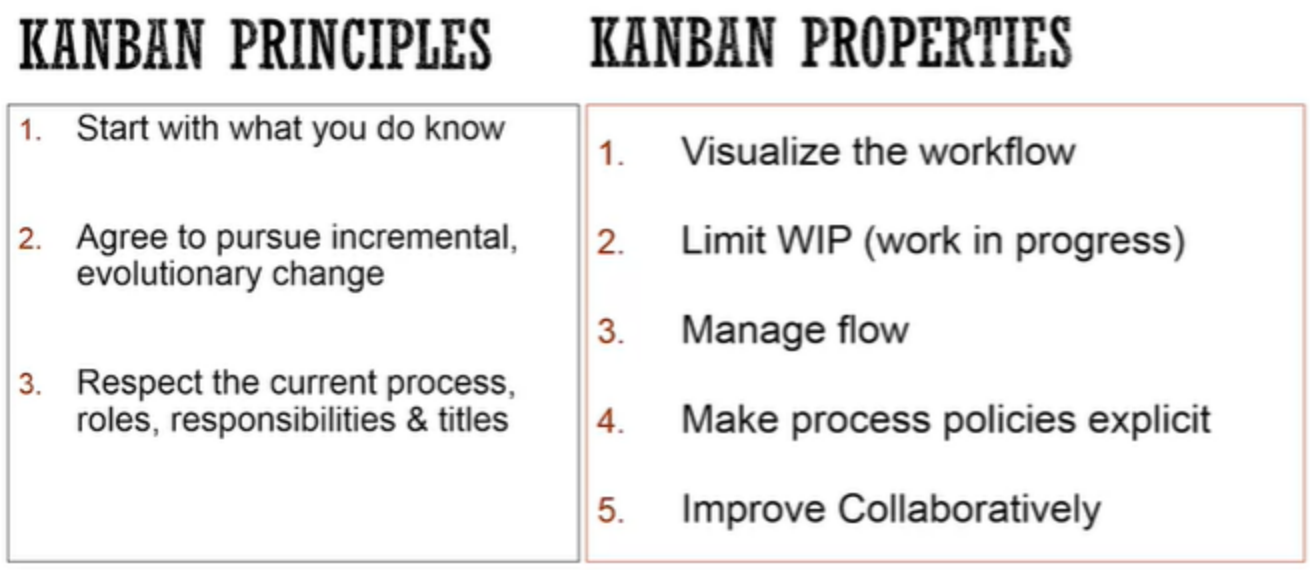
How do you optimize the whole?

Limits to Growth

Kanban: Helps you identify issues in your software development process flow by limiting work in progress

Week 2

Kanban for software development



Lean Metrics to Improve Flow

Key Takeaway: Identify the queues in your workflow so you can see where work could get stuck in your process. Track the amount of work you have in queues and try to minimize it, relative to the total WIP in the system. Shorter queues lead to shorter wait times and lower overall cycle time.

https://www.planview.com/resources/guide/what-is-agile-program-management/lean-metrics-improve-flow/

Value Stream Mapping

Map out the activities of your process you want to analyze

Go to the place where work is happening

Identify Value Added work, Non-value Added work

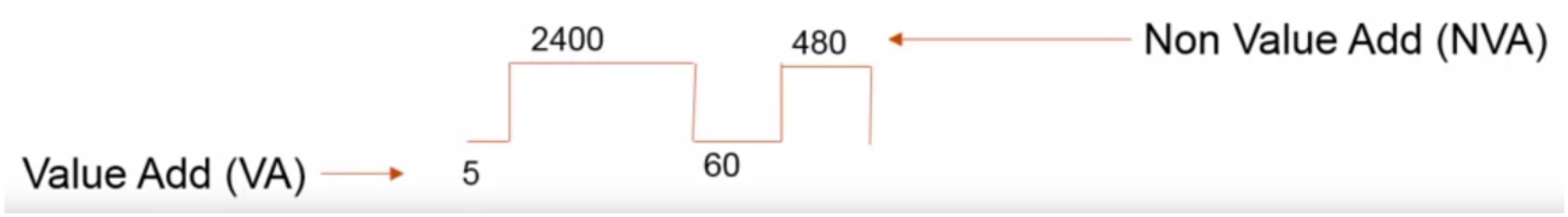
Who are involved, What is involved and How long it takes

Syntax & terminology

Value stream →
+ A = 5 min
+ B = 5 days (non value add)
+ C = 1 hr
+ D = 1 day (non value add)

In this case, we will show time in minutes

1 day = 8 hrs
1 hr = 60 m
→ So 5 days in minutes = 5 days * 8 hrs * 60 min = 2400 min



Process Cycle Efficiency = Value Added Time / Cycle Time

`Value Added Time` Time spent in doing things that adds value for the customer

`Non Valued Added Time` Time spent in doing things that didn't add value for the customer

Cycle Time = Total time taken for value stream

Process Cycle Efficiency =

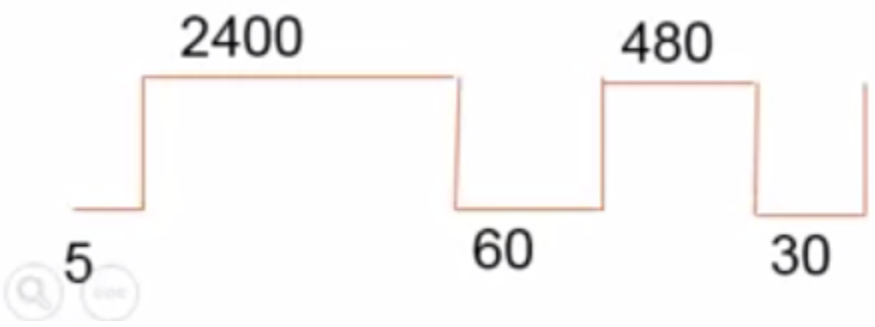
$$\frac{5+60}{5+60+2400+480} = 2.2$$

Case study

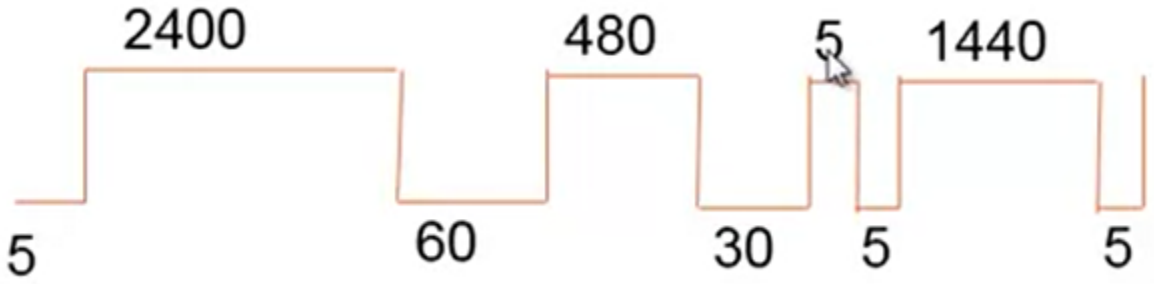
- ABC HealthCare Corp: 15,000 Employees
- IT Department: 2,000 Employees
- Database Support Group (DSG) (30 employees)
- 120 application development teams
- DSG responsible for all database related activity required by any of the 120 application development teams

Value stream map

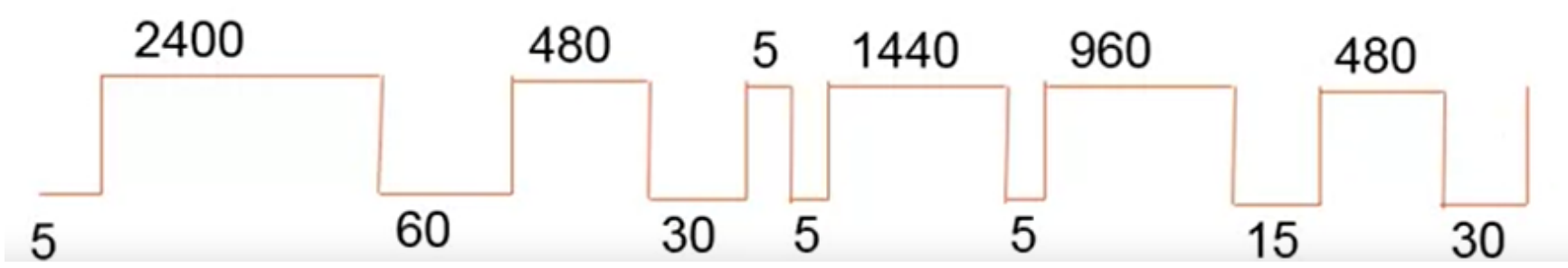
- Developer submits a ticket to get database work done (5 min)
- Ticket waiting to be looked at and prioritized (5 days)
- Key Managers from different departments attend weekly prioritization meeting to finalize prioritization collectively (1 hr)
- Prioritized items wait for being assigned to a database administrator (1 day)
- Prioritized item is finally assigned to a database administrator (30 min)

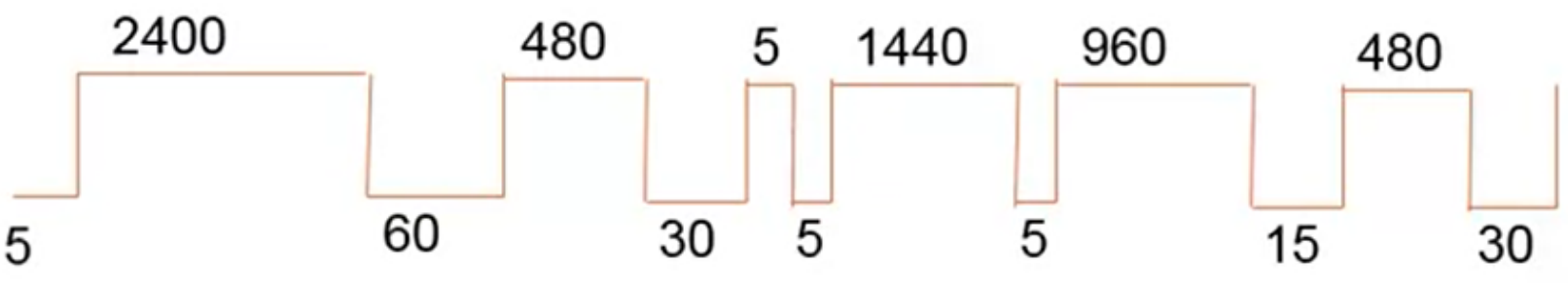


- Data administrator sends an email to request additional information in a template specifically designed for that type of request (5 min)
- Developer provides the information requested (5 min)
- Request wait in the queue to be picked up again (3 days)
- Database administrator picks request and setup meeting with developer for additional information (5 min)



- Developer waits for the meeting (2 days)
- Developer and Database administrator meets to finalize the exact requirement (15 min)
- Database administrator puts it back into queue to be worked upon (1 day)
- Database administrator works on the request and informs the developers once done. (30 min)





Process Cycle Efficiency = $\frac{\text{Value Added Time}}{\text{Total Cycle Time}}$

Value Added Time = 5 + 60 + 30 + 5 + 5 + 15 + 30 = 150

Total Cycle Time = 5 + 60 + 30 + 5 + 5 + 15 + 30 + 2400 + 480 + 5 + 1440 + 960 + 480 = 5915

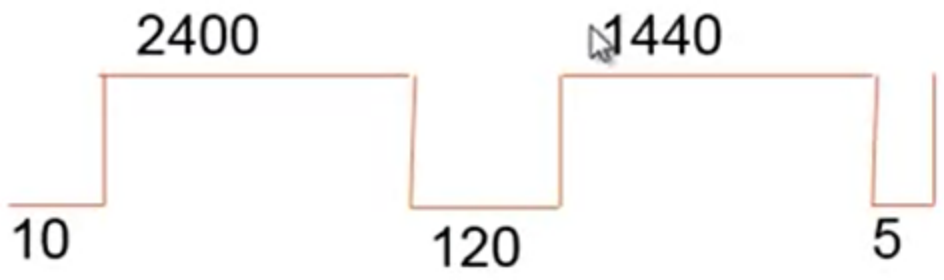
Process Cycle Efficiency = 150 / 5915 = 0.025 = 2.5%

Can we improve?

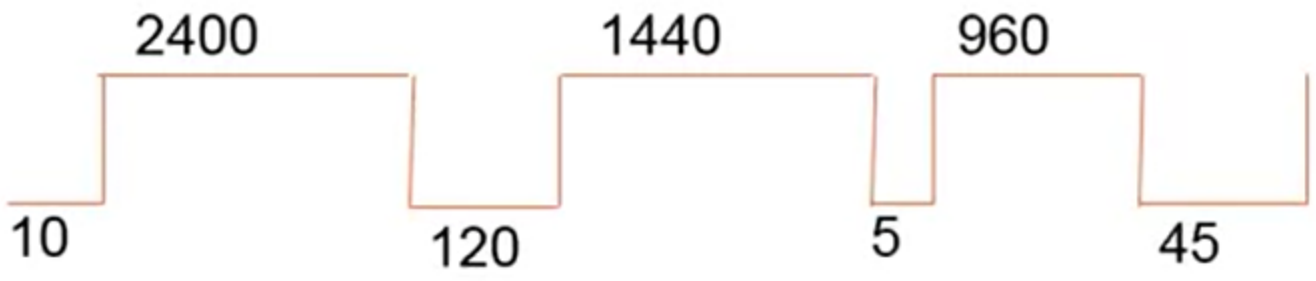
- Developer submits a ticket to get database work done. Developer selects the type of request and fill appropriate info needed for that type of request (10 min)

- Developer submits a ticket to get database work done. Developer selects the type of request and fill appropriate info needed for that type of request (10 min)
- Ticket waiting to be looked at and prioritized (5 days)
- Key Managers from different departments attend weekly prioritization meeting to finalize prioritization and assign it to a database administrator (2 hrs)

- ~~Data administrator sends an email to request additional information in a template specifically designed for that type of request (5 min)~~
- ~~Developer provides the information requested (5 min)~~
- Request wait in the queue to be picked up again (3 days)
- Database administrator picks request and setup meeting with developer for additional information (5 min)



- Developer waits for the meeting (2 days)
- Developer and Database administrator meets to finalize the exact requirement AND works on the request in the meeting itself. (45 min)
- ~~Database administrator puts it back into queue to be worked upon (1 day)~~
- ~~Database administrator works on the request and informs the developers once done. (30 min)~~



Process Cycle Efficiency = Value Added Time / Total Cycle Time

Value Added Time = 10 + 120 + 5 + 45 = 180
Total Cycle Time = 10 + 120 + 5 + 45 + 2400 + 1440 + 960 = 4980

Process Cycle Efficiency = 180 / 4980 = 0.036 = 3.6%

Assignment: Value Stream Map

In this assignment, you will be asked to create a value stream map for the scenario below. You will then use it to compute the process cycle efficiency and make recommendations for improvements.

Wise Software Corp (WSC), a short-term contracted software development company, has the following process for serving its clients.

Clients first create an account by providing basic profile information (10 min)

Clients wait for the approval of their create account request (avg. 12 hrs)

Company's system runs in the night and sends email confirmation email (1 min)

Email sits in Client's email box (avg. 12 hrs)

Client confirms their account by clicking on a link in an email. (1 min)

Client submits a new project request on WSC's website portal (1 hr)

Request is waiting to be assigned to an account manager (12 hrs)

Company's Sr Account Manager looks at the pending requests every day and assign it to one of the account manager (1 min)

Request is waiting for an account manager to looks at the request (12 hrs)

Account manager contacts the client to set up a meeting to get requirements (30 min)

Client and Account Manager waiting for the meeting (72 hrs)

Account Manager and Client meet to outline requirements(4 hrs)

Account Manager puts the request in queue to document (24 hrs)

Account Manager writes up requirement and sends this information to Company's Solution Architect (30 min)

The project is waiting for a Solution Architect to provide input (48 hrs)

Solution Architect looks at and provides feedback to Account Manager (1 hr)

Solution Architect feedback waiting to be read by Account Manager (24 hrs)

Account Manager sets up another meeting with Client to discuss architecture and timeline (30 min)

Client and Account Manager waiting for the meeting (72 hrs)

Account Manager and Client meet to approve architecture (1 hr)

Account Manager sends the project request to the development queue (30 min)

Project request waits in development queue (120 hours)

Project team completes the development and assigns it to a testing team (72 hrs)

Project request waits in testing team to work on it (120 hrs)

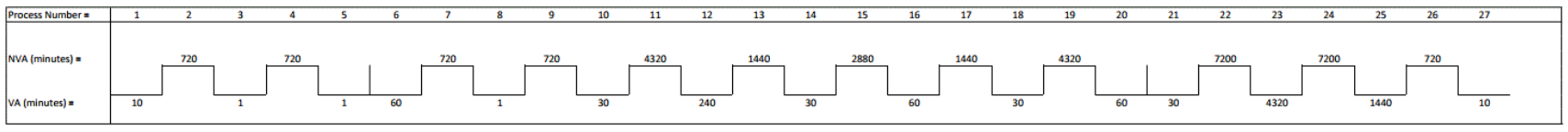
Testing team tests and approves it for release (24 hrs)

Project waiting for Account Manager to send the request to Client (12 hrs)

Account Manager sends the completed project deliverables to Client. (10 min)

Client uses the deliverables. (0 min)

Please create a Value Stream Map for the process of Wise Software Corp's software development process.



Please calculate the Process Cycle Efficiency for this Value Stream? Please show your calculations.

Value Added Time = 6323 minutes

Non Value Added Time = 32400 minutes

Cycle Time = 6323 + 32400 = 38723 minutes

Process Cycle Efficiency = Value Added Time / Total Cycle Time

Process Cycle Efficiency ⇒ 6323 / 38723 ⇒ 16.33%

Make a recommendation for improving this process. Please specify what impact it will have in terms of VSM step and Process Cycle Efficiency. Please calculate the new Process Cycle Efficiency if recommendations are implemented (HINT: For recommendations, think about merging steps, removing unnecessary steps, or recommending alternatives to improve some of the process)?

1. Were there recommendations to merge steps (signup and submitting a project)?

Steps that will be merged were specified and the new Process Cycle Efficiency was calculated.

2. Was there a recommendation to take Architect into the requirements meeting and eliminate a bunch of steps?

Recommendation was made and included all the steps that it will eliminate and the new Process Cycle Efficiency was calculated.

View the entire process flow from start to finish and create a plan to optimize efforts to ensure that the company achieves the desired results. By illustrating the current state, the value stream map lets you know where the current gaps are.

Kaizen

Kaizen is a very simple technique used for continuous improvement.

Learning objectives

Overview of Kaizen

- Office
- Factory
- Healthcare

Kitchen Kaizen!

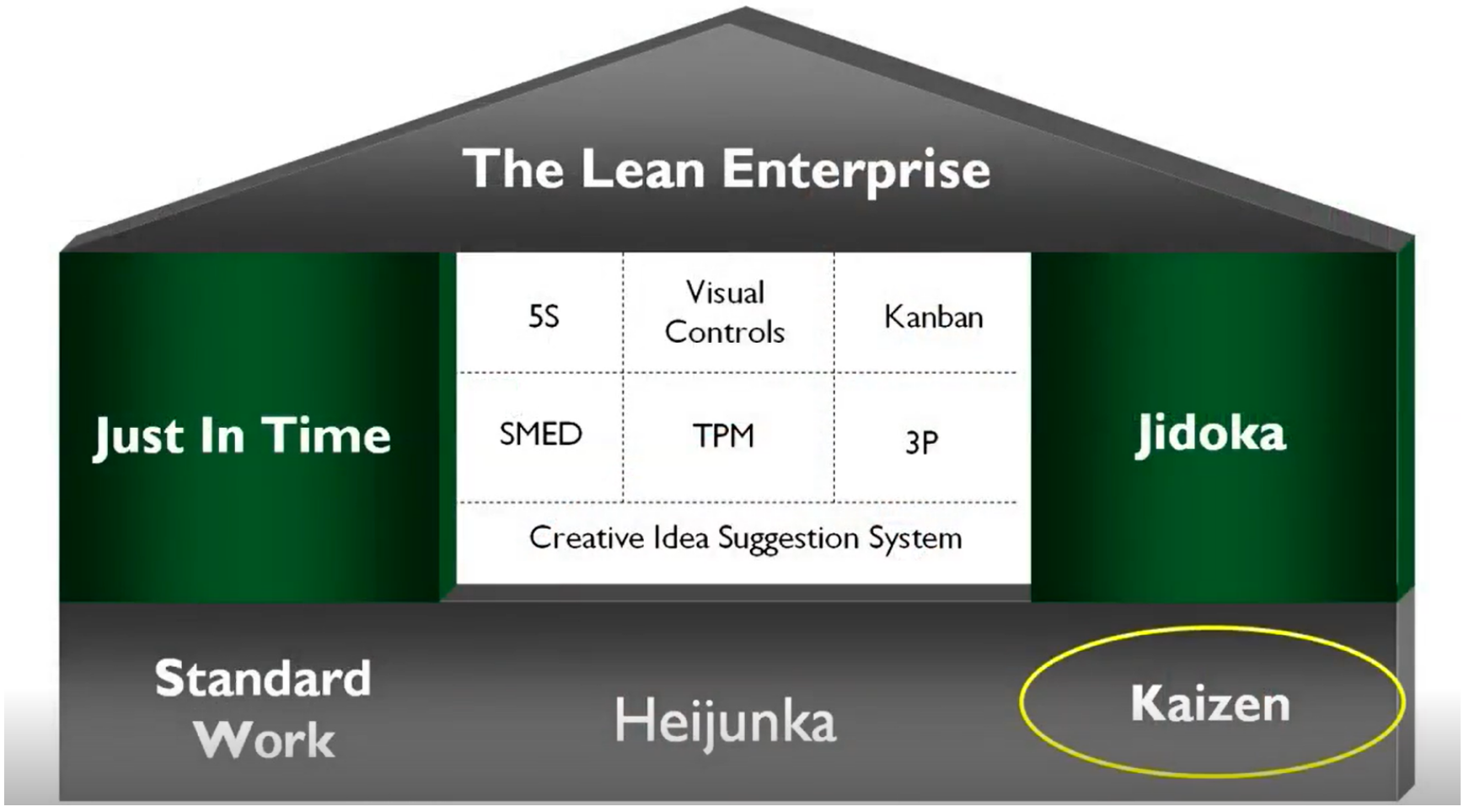
What is Kaizen?

Mean “change for the better”

Sense of breaking down the current process, removing the unnecessary parts, putting it back together in an improved manner

Take a look at the current process, break it apart and put it back together again as such the result shoud be an improved process that fully utilizes all the experience and skill of the people involved

Kaizen is also a cornerstone of the lean enterprise as it works together in harmony



The “3 gen” principle / 3 actuals

1. When we do Kaizen, we must go the genba

gen-ba or gem-ba = actual place (the place the work is done)

Might be the factory floor, a construction site, the operating room in a hospital

2. Look at the actual parts

Gen-butsu = actual parts

Instead of looking at a flow chart, spending time walking and experiencing the process for ourselves.

3. Get facts that either prove or disprove out ideas in a non emotional manner

genjitsu = Get the facts

Kitchen Kaizen

“Earn the right to go into the kitchen”

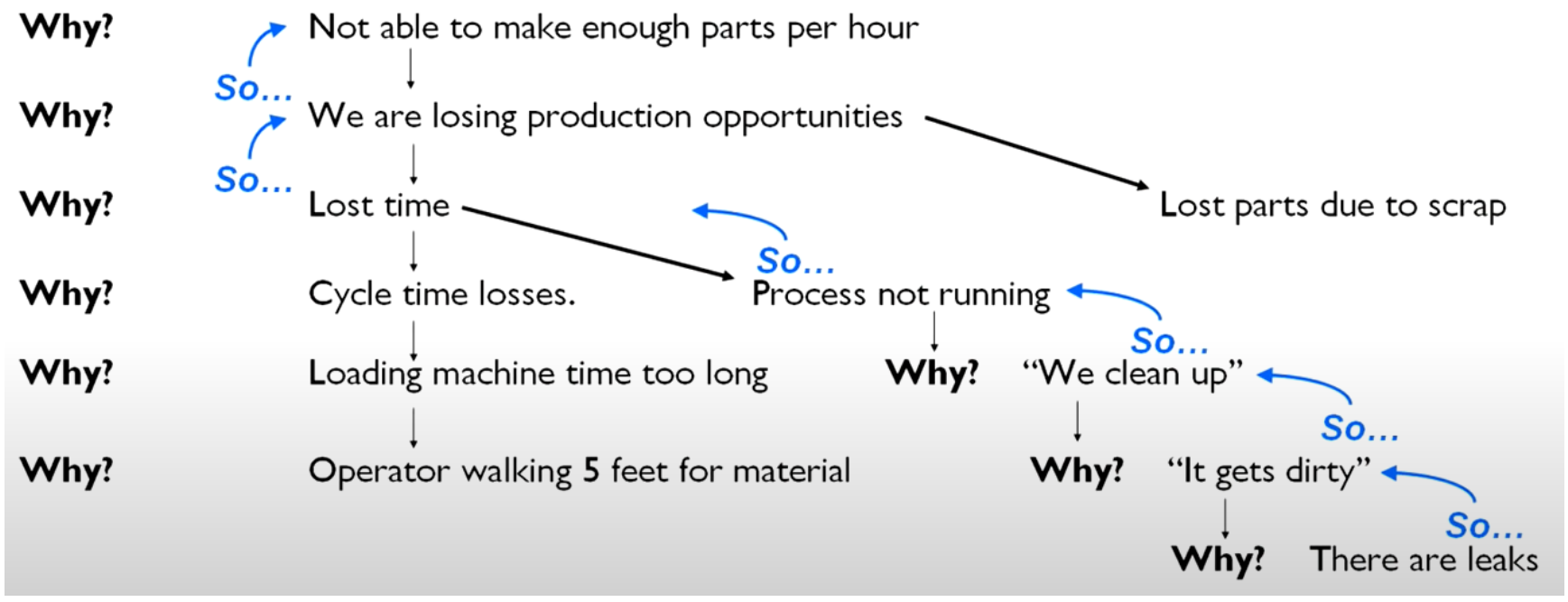
This means that we can’t simply barge into someone’s area and start asking questions or changing things.

In Kaizen, we must earn the right by showing respect and developing trust first

Kaizen Examples

The 5 Whys

The 5 why is a problem solving methods used to explore the cause and effect relationship of a particular problem



"... the basis of Toyota's scientific approach ... by repeating why five times, the nature of the problem as well as its solution becomes clear". - Taiichi Ohno

Problem The milk is spoiled

Why?

– We didn't drink all the milk cartons before expiration

Why?

– We had too many cartons in the fridge

Why?

– We bought more cartons than we needed

Why?

– There was a sale on milk and we tried to save money

Two primary techniques are used to perform a five whys analysis: the **fishbone** (or Ishikawa) diagram. a **tabular** format.

Week 3

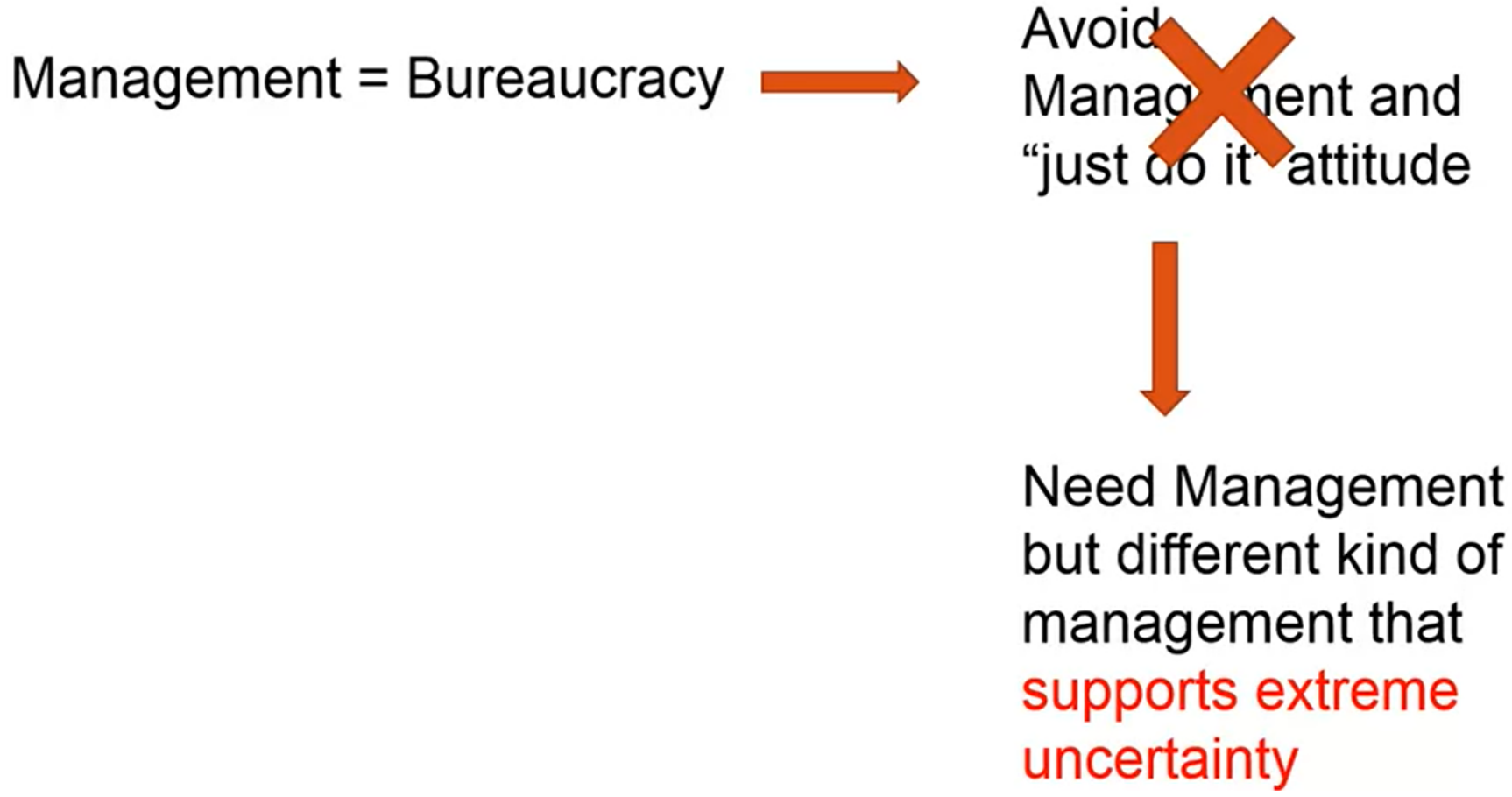
Lean Startup Principles

Origin of lean startup

A human institution designed to create new products and services under conditions of extreme uncertainty

Principle 1: Entrepreneurship is management

Management = Bureaucracy



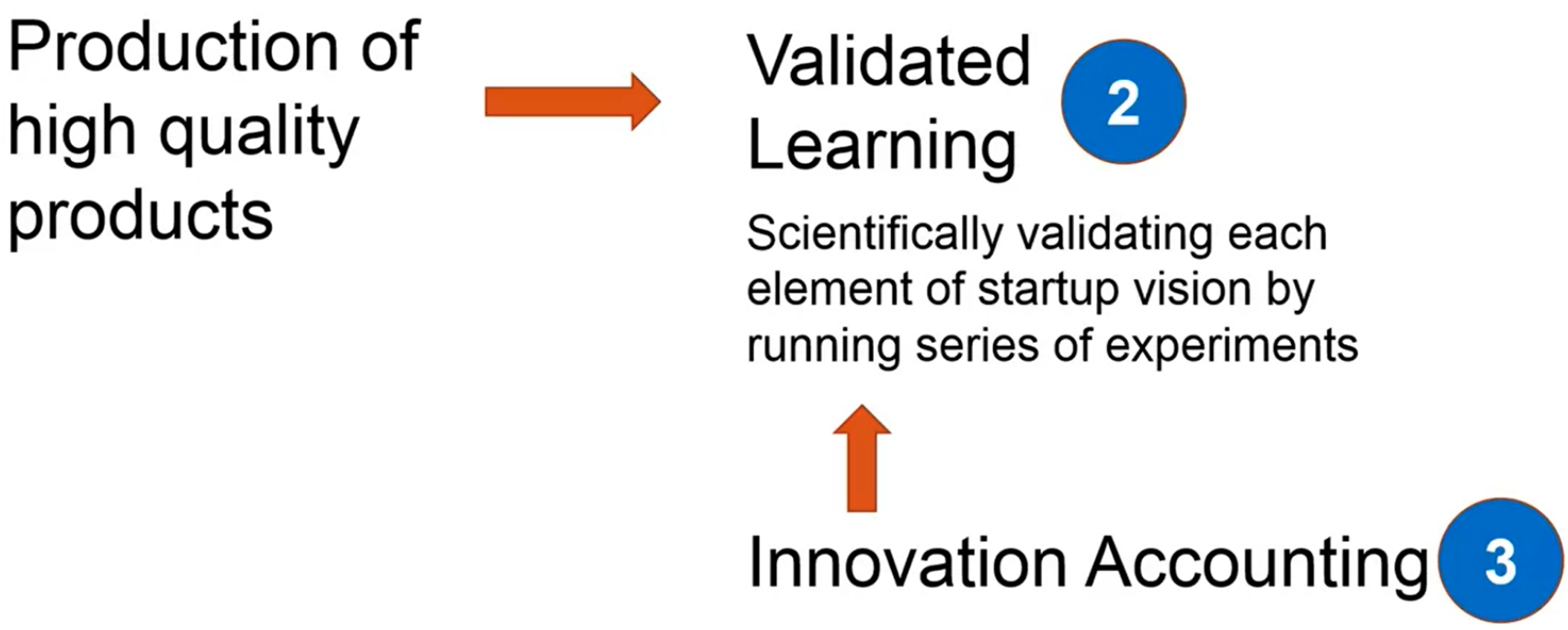
Measure progress

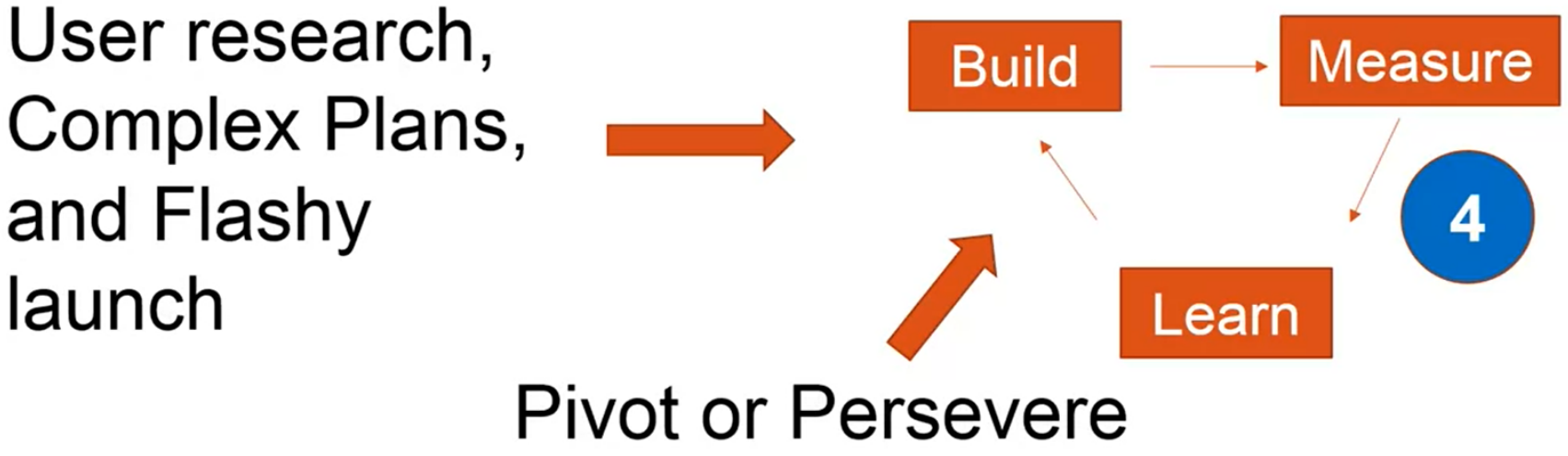
Production of high quality products → Validated learning

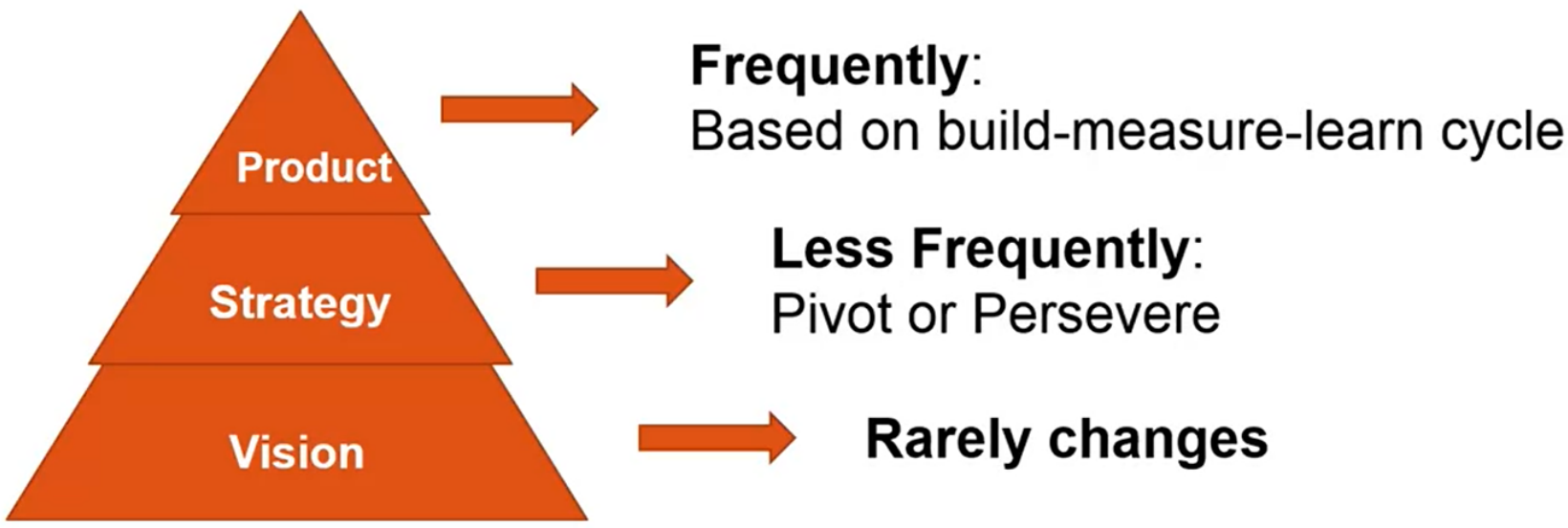
Principle 2: Validated learning

Scientifically validating each element of startup vision by running series of experiments

Principle 3: Innovation accounting







Principle 4: Build - Meausre - Learn approach

Applicable to only start-ups?

Equally applicable to enterprise

Small or latge, profit or non-profit, commerical or government organizations

→ Lean start-up ecosystem

Equally applicable to enterprise

small or large, profit or non profit, commercial or government organizations

→ Lean Startup Ecosystem

Trying to *create new products and services under conditions of extreme uncertainty*,

Principle 5: Entrepreneurs are everywhere

Entrepreneurs could be in an organization in a big enterprise

→ Intrapreneurs

Intrapreneurs → Entrepreneurs that are trying to build a Start-up inside an established organization

Lean Startup (Validated Learning and Build-measure-learn)

The idea behind is hoping you learn faster about market and user need

Lots of software development model focus on delivery

This model focus a lot on understanding or fast learning about real user need

What does it look like?

Validating learning

> We must learn what customers really want, not what they say they want or what we think they should want. We must discover whether we are on a path that will lead to growing a sustainable business." - Eric Ries

Complete the cycle as quickly as possible

Assumption

What metric can I measure to validate or invalidate my assumption?

What experiment can I give, can I do?

Innovation Accounting

Accouting for start-ups

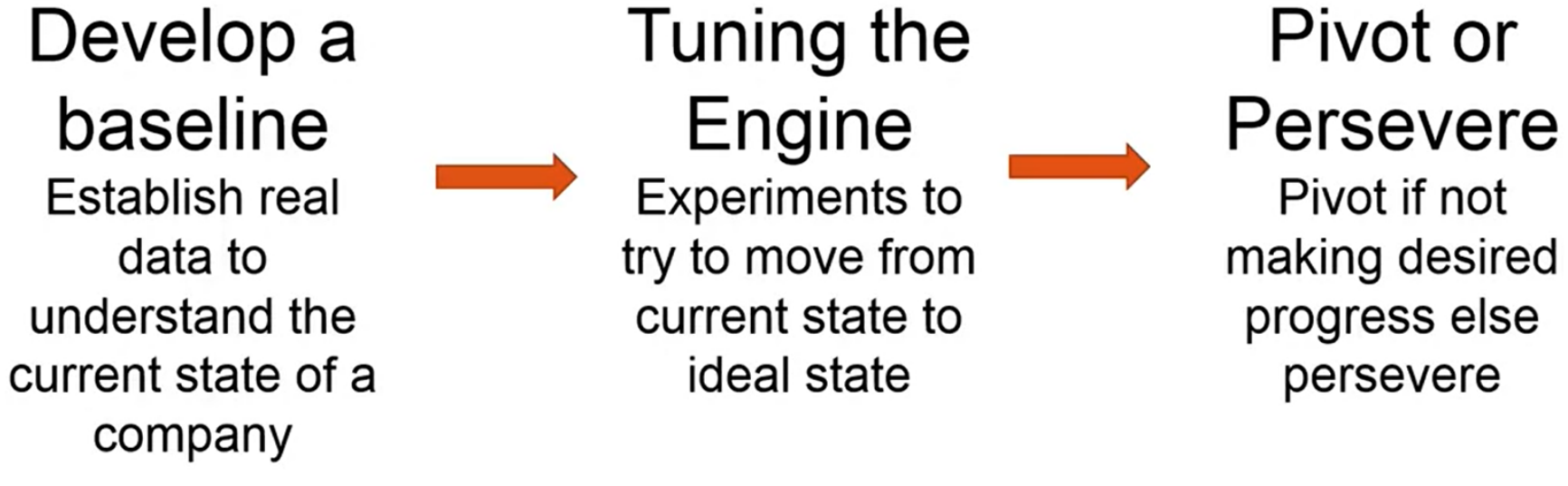
Do you really need accounting?

Wouldn’t accounting be applicable once the start-up is successful and established?

Yes, but different kind of accounting?

It helps you make right decision at the right time

3 learning milestones



Develop a baseline

Prototype

- Test multiple assumptions at a time
- One assumption at a time
  + Try the riskiest assumption first - These are also called “leap-of-faith” assumptions

Smoke test

- No real product - customers are given an ability to pre-order the product that is not yet built
- This validate if customer are interested in trying product

Tuning the engine

Activities (product dev, marketing or other activities) targeted towards improving the driver of growth model

The 3 A’s of metrics

Actionable

- Must demonstrate the cause and effect
- Should not be a vanity metric
  + Vanity metrics can lead to false conclusion and can mask the failure

Accessible

- Easy to understand
- Easy access to report/data
  + Metrics part of the app

Auditable

- Data is credible
- Manually testable

Pirate metrics (AARRR)

Pivot or persevere

Startup’s runway

- Amount of time remaining to eight lift-off or fail
- Amount of money remaining / monthly burn rate
- Extend by getting more money or cut costs

Startup’s runway - true definition

- Number of pivots remaining
- Increase your runway by reducing time for each pivot

Pivots require courage

- Vanity metrics
- Incorrect hypothesis
- Afraid to not get the chance to prove

Pivot or persevere meeting

- Schedule in advance

Types of pivots

Zoom in

Zoom out

Platform

Customer segment

Customer need

Channel of delivery

Technology

Week 4

Design thinking

Origin of design thinking

Design – way of thinking
1969, Herbert Simon

→

Method of creative action
1991, Rolfe Faste

→

IDEO: Commercial application of Design Thinking
1991, David Kelley

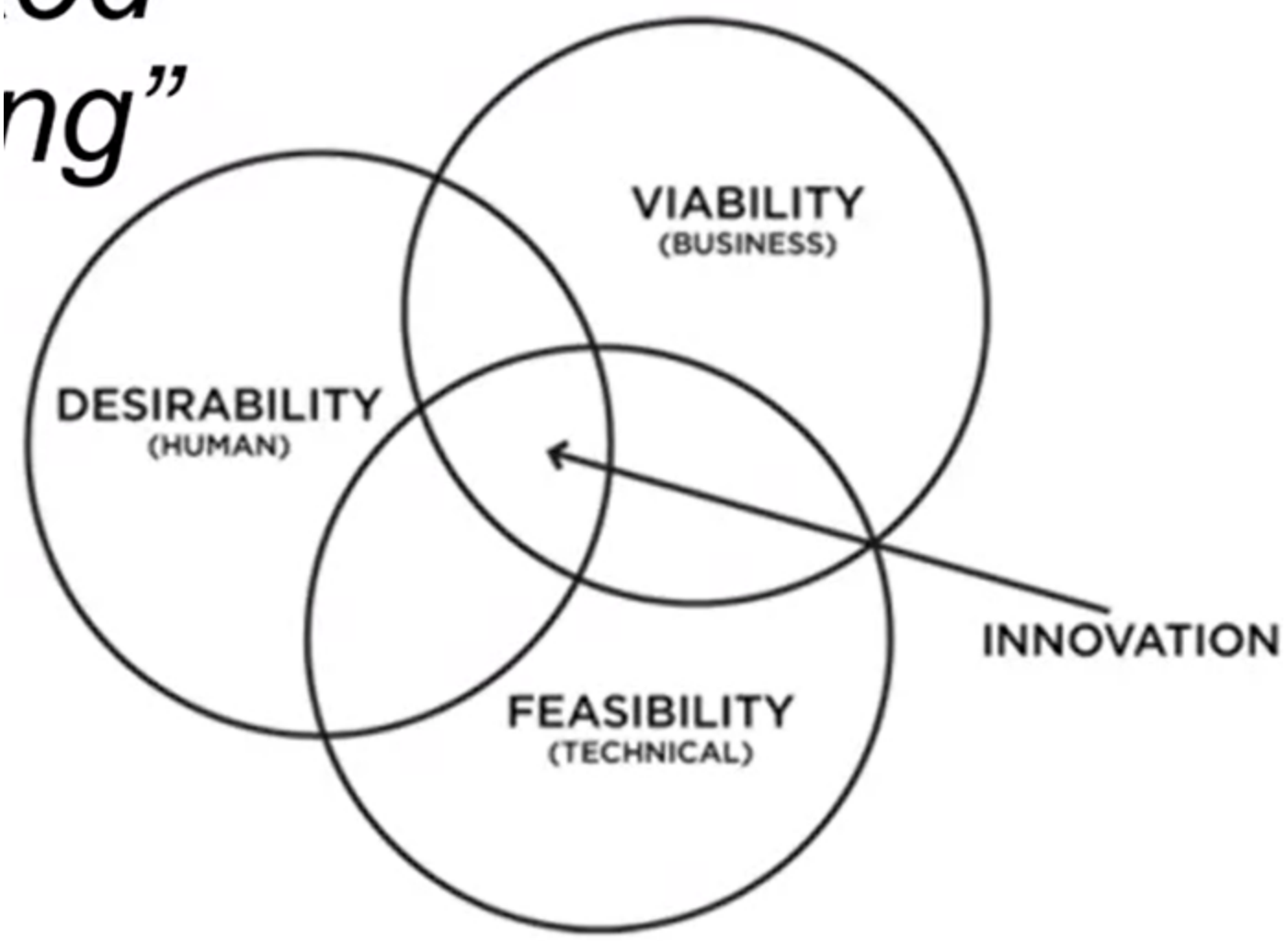
What is design thinking?

“Design thinking is a process for creative problem solving”

“Methodology for creative and practical wicked problem solving”

“Design thinking is a human-centered approach to innovation that draws from the designer’s toolkit to integrate the needs of people, the possibilities of technology, and the requirements for business success” - Tim Brown, CEO, IDEO

ed
ng”



Step of design thinking

Empathize

- Develop a deep understanding of the challenge

Define

- Clearly articulate the problem you want to solve

Ideate

- Brainstorm potential solutions
- Select and develop your solution

Prototype

- Design a prototype (or series of prototypes) to test all or part of your solution

Test

- Engage in a continuous short-cycle innovation process to continually improve your design

Cross functional team – not a waterfall – go back and forth between steps

Empathize

Why?

- Prevent bias/filter
- They are not our problems
- Identify real problem

How?

- Go where users are
- Talk directly to customers and users
- Watch them work → Work with them
- Identify pains, needs etc

Define

Why?

- Keep focus
  + Define target group
  + Define the real problem

How?

- Put everything you observed on board
- Share stories
- Distill learning
- Define the user/target group
- Define / Redefine the problem statement
- Focus on one or few problems

Ideation

Why?

- Don’t lock-in too early
  + Defer commitment, amplify learning
- Find innovative solution

How?

- Come up with multiple solutions - Lots of them
- Go beyond rational thinking
- Two rules
  + Visualize
  + Don’t sit
- Select few

Prototypes and Test

Why?

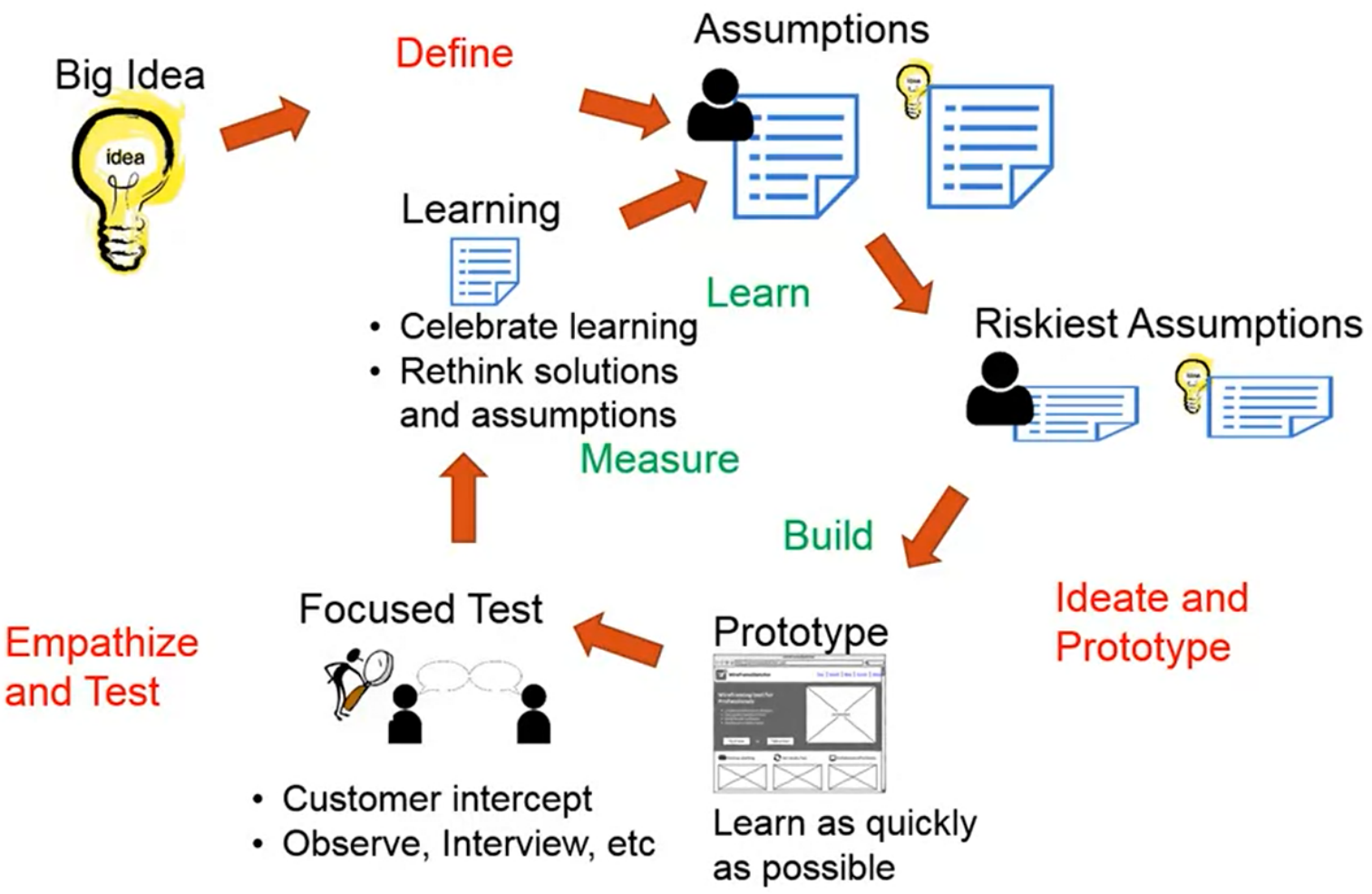
- Validating ideas (quick and inexpensive)

How?

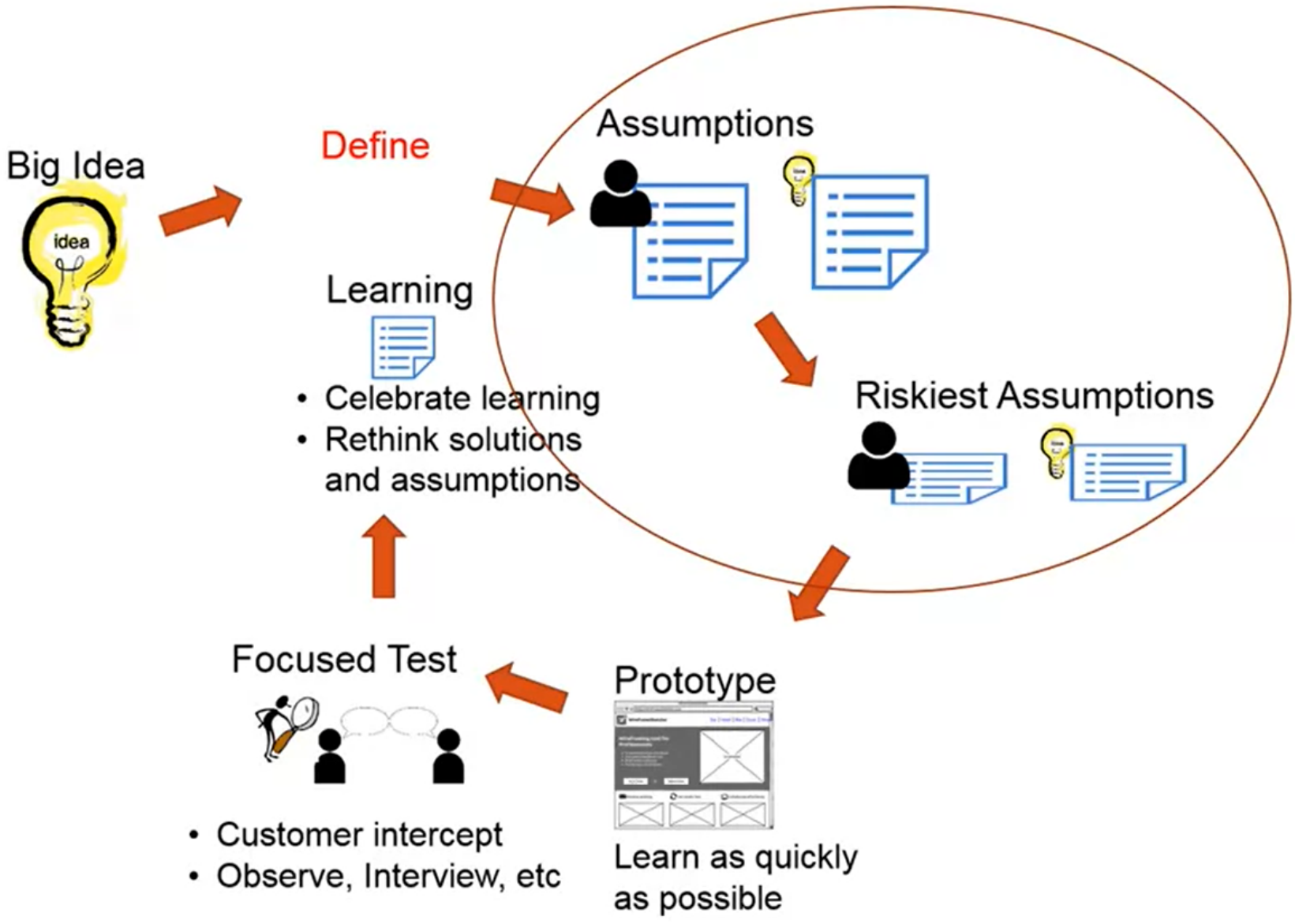
- Physical, interactive prototype
- Right fidelity
- Prototype with defect is ok - main idea is to validate if it sovles the problem

- Don’t defend the idea
  + Note down and move on

Lean startup with Design thinking



Identifying and Classifying Assumptions



Listing assumptions

Collective brainstorm (tech, business, stakeholders)

What are we assuming about our users/customers?

What are we assuming about our solutions

What must happen for our solution to be succesful after it is released?

- Will users engage in activities we are expecting them to?
- Will users recognize the value?
- Will users be able to use the solution?
- Will users have what they need (context) to use the solution?

Classifying assumptions

Two dimensions

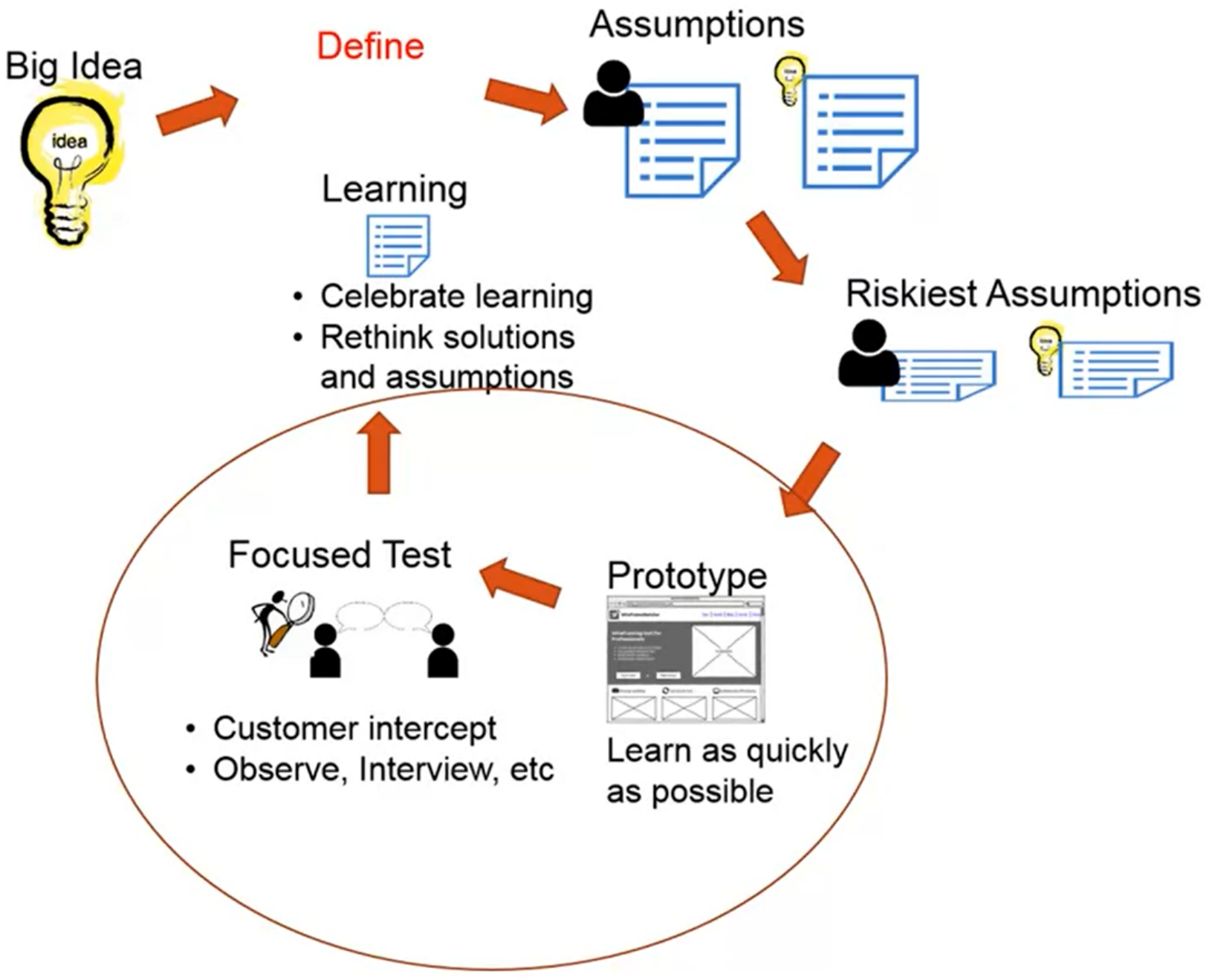
- Probability that it will be wrong
- Impact on our solution if it is wrong

Four variations

- Low probability that it will be wrong, Low impact on solution
- Low probability that it will be wrong, High impact on solution
- High probability that it will be wrong, Low impact on solution
- High probability that it will be wrong, High impact on solution

Start with “High probability that it will be wrong, High impact on solution”

Prototype and Test



Listing potential ideas

Collective brainstorming (Tech, Business, Stakeholders)

Customers intercepts - go where users are

Types of tests

- Interview
- Pitch your solution and get a reaction
- Prototype: different degree of fidelity - paper to real application/real data
- Observe
- Entice user on existing platform - Feature-Fake

Classifying MVP tests/experiments

Twp dimensions

- Cost/Effort of conducting test
- Quality of data collected from the test

Four variations

- Low cost/effort, Low quality of data
- Low cost/effort, Highquality of data
- High cost/effort, Low quality of data
- High cost/effort, High quality of data

Start with “High cost/effort, High quality of data”

EXAMPLE – DIGITIZE BOOKS

1. It is a problem for students to carry around books

2. Tablets will provide better reading experience
    1. Test: Observe and Interview (high cost, high quality)
    2. Campus Survey (low cost, low quality)

3. Students will be willing to pay for tablet cost.
    1. Campus survey (low cost, low quality)
    2. A video and signup link on student portal (low cost, high quality)

4. Prospective students will have favorable rating for college if we provide digitized books.
    1. A/B Testing on college admission form (high cost, high quality)
    2. Survey during the Open House for prospective students (low cost, low quality)

Learning from the test

Consolidating / Classifying learning

Collective brainstorming (Tech, Businesss, Stakeholders)

Affinity mapping

Consolidate learning

- Invalidated assumptions
- Validated assumptions
- New ideas/questions/assumption

Assignment: Applying design thinking

In this assignment you will apply some of the Design Thinking techniques (identify assumptions and risks, select the riskiest assumptions, and write MVP tests for the riskiest assumptions) to a fictitious situation. Feel free to make reasonable assumptions about user needs and additional functionality required to satisfy user needs.

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/2f3b70d3-5014-4fe3-b477-b9568b66d7c6/Case-Study-Remote-Deposit-Capture-Project.pdf

Identify some of the assumptions made about the success of remote deposit capture functionality. Assess the risk of each assumption affecting our product by classifying the assumptions into following four categories:

High impact if wrong, High Probability of it being wrong

High impact if wrong, Low Probability of it being wrong

Low impact if wrong, High Probability of it being wrong

Low impact if wrong, Low Probability of it being wrong

For each assumption, provide the reason why you categorized the assumption into the particular category.

Assumption #1 - Clients will actually buy scanners to deposit checks.

- Low impact since we also have mobile phone option for scanning.
- High Probability that this is wrong assumption based on the current industry trend.

Assumption #2 - Blue Bank's existing clients want this feature.

- Low impact since if this is wrong, the bank can still benefit from new customers.
- Low probability of it being wrong since it will make check deposits easy for existing customers and many current banks already offer this service.

Assumption #3 - Potential customers will open an account with Blue Bank if we launch this feature

- High impact since, if this assumption is wrong, the Bank does not benefit from this project
- Low probability of it being wrong based on the current industry trend.

For any three assumptions you identified, please list the MVP Tests or Minimum Viable Experiments you can conduct to validate the assumptions? Please categorize your MVP Tests according to the following categories. It is okay to have 1 or 2 categories with no tests.

High Cost, High Quality Data (tests that will be costly to conduct, but will result in high quality data)

High Cost, Low Quality Data (tests that will be costly to conduct and will result in low quality data)

Low Cost, High Quality Data (tests that will be inexpensive to conduct, but will result in high quality data)

Low Cost, Low Quality Data (tests that will be inexpensive to conduct and will result in low quality data)

Did the learner provide good MVP tests for the selected assumptions?

Examples of good MVP tests:

- A link on the website to enroll in the remote deposit functionality (Low Cost, High Quality Data)
- Link on the website to get more information on an app to deposit checks remotely (Low Cost, Low Quality Data)
- A survey to ask for preferred method of deposit checks (Low Cost, Low Quality Data).
- A survey to find out of existing clients have a scanner, and, if not, would they buy one for depositing checks (Low Cost, Low Quality Data).