



# CERTIFIED SCRUM PRODUCT OWNER TRAINING



# Peter Saddington MDiv, CST

- 3X Startup Founder, Author, Agile Trainer, and Brand Consultant
- 20+ years of software development experience
- Social Scientist - M.A. Counseling, M.A. Education, M.Div. Theology
- Peter's Community - [www.vchunting.com](http://www.vchunting.com)
- *The Agile Pocket Guide - A Quick Start to Making Your Business Agile* (2012)
- *Gravity - How to Create an Irresistible brand, Unlimited Deal Flow and Infinite Opportunities* (2020)

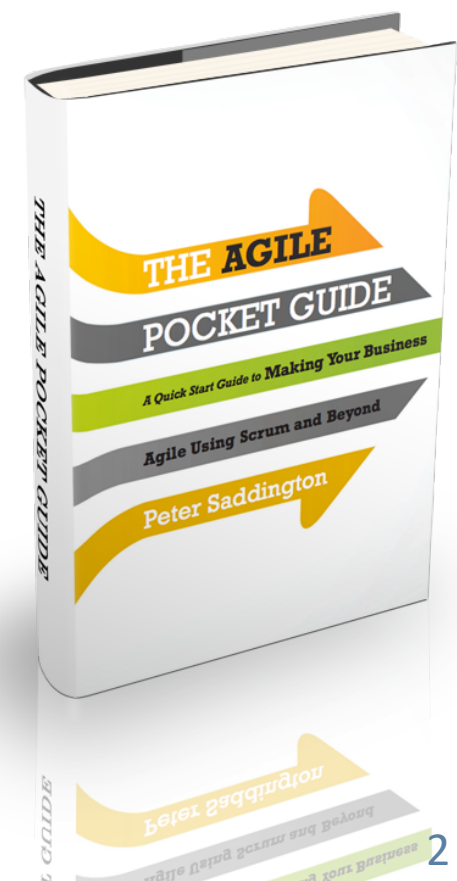
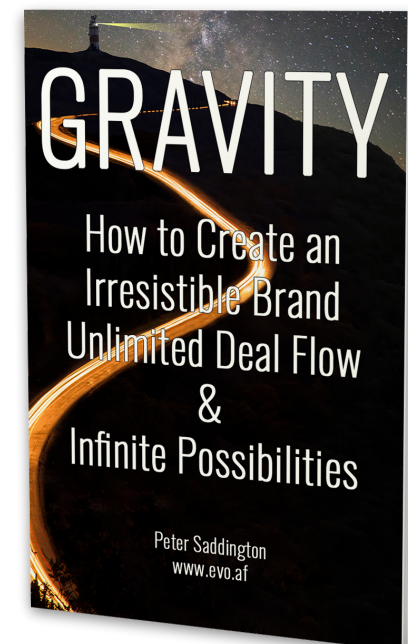


Email: [agilepeters@gmail.com](mailto:agilepeters@gmail.com)

Web: [agileforall.com](http://agileforall.com)

Twitter: [@AgilePeter](https://twitter.com/AgilePeter)

Book: [vchunting.com](http://vchunting.com)





# The Agile Manifesto Principles

**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

# Plan Driven Development - Not a Proposal

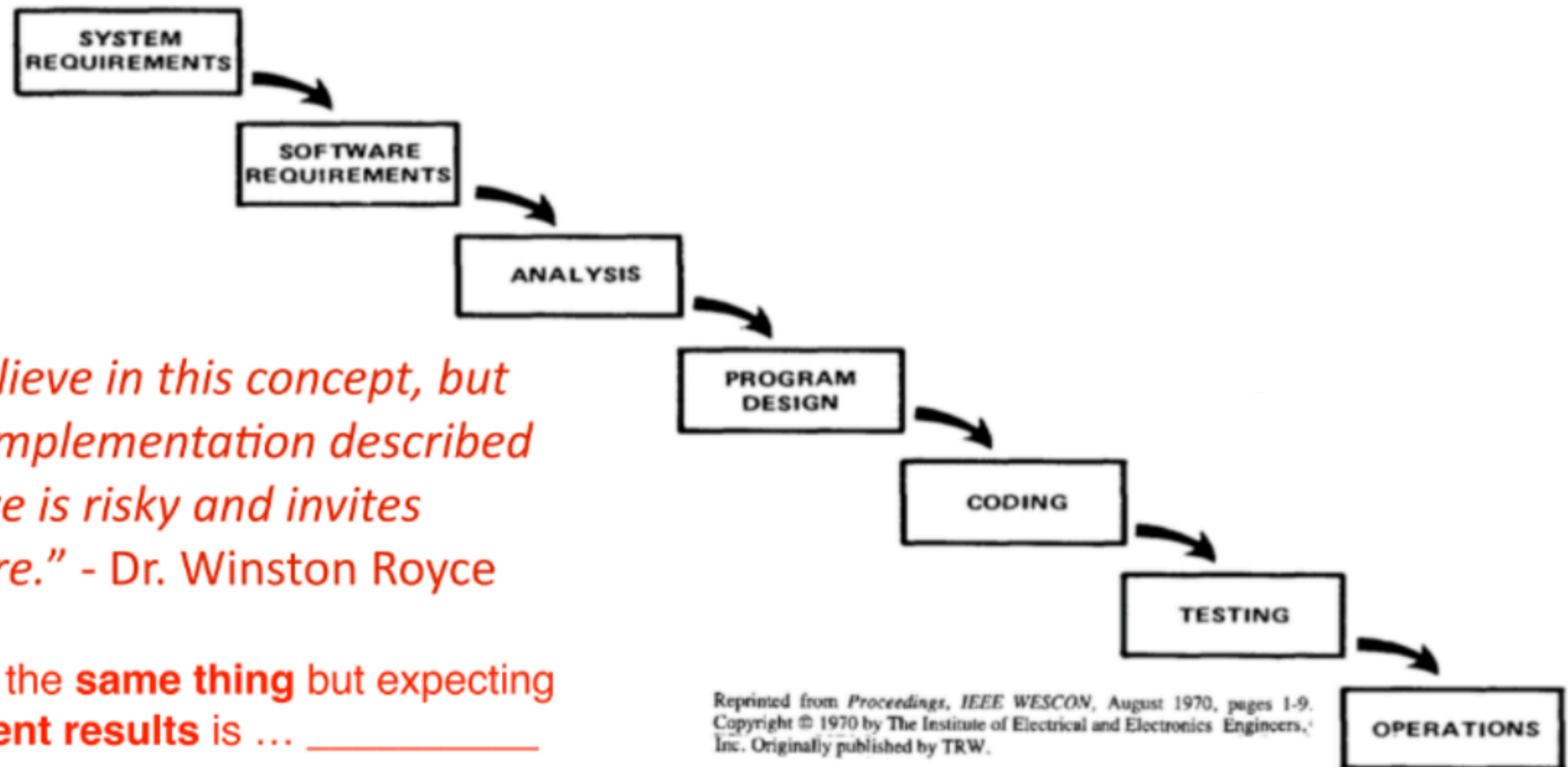
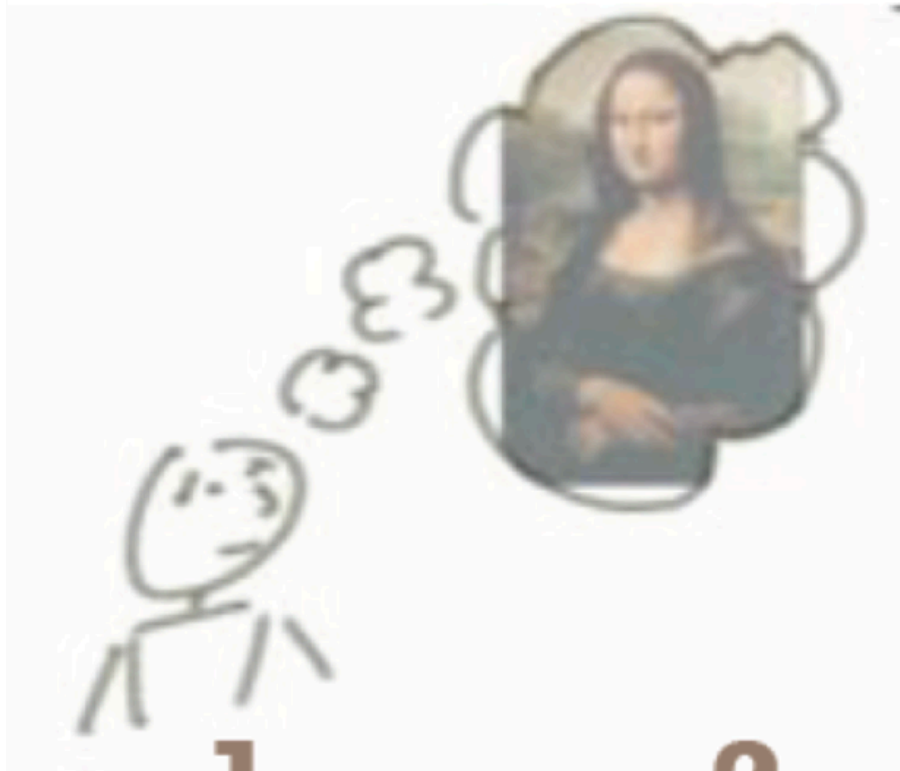


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

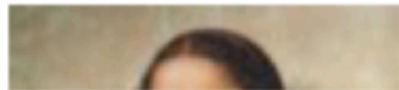
# Incremental Product Development

Incrementing calls for a fully formed idea.

And, doing it on time requires dead accurate estimation.



1



2



3



4



5





## Iterative + Incrementing Builds a Rough Version, Validates it, then Slowly Builds up Quality



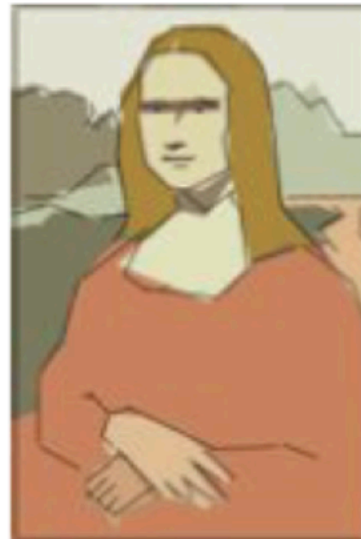
1



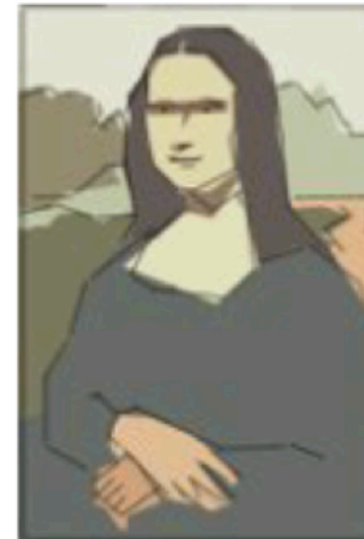
2



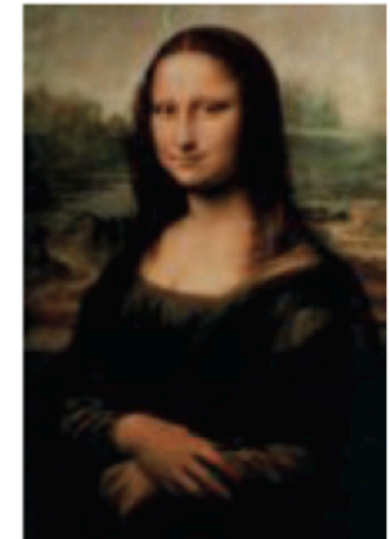
3



4



5



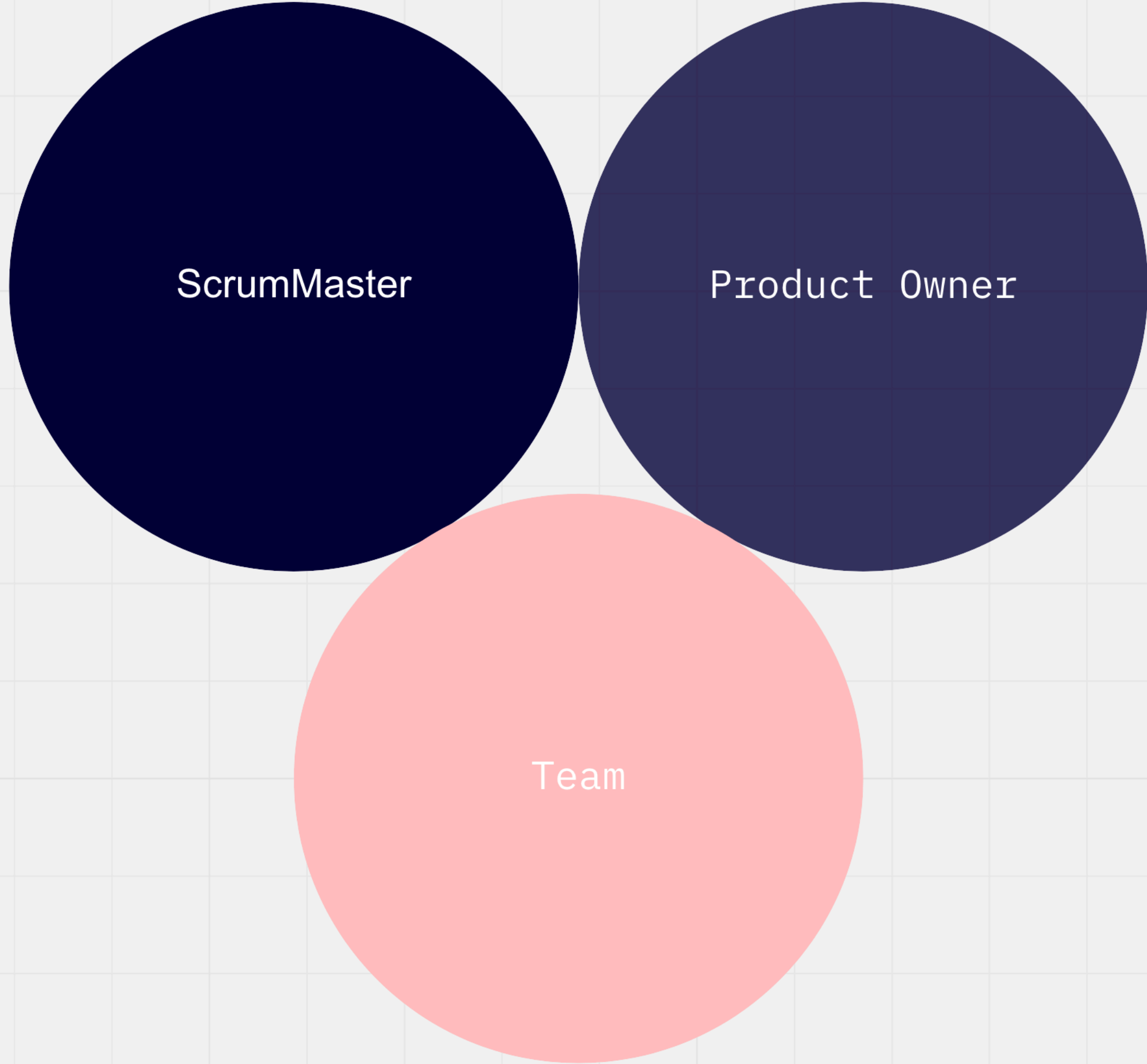
A more iterative allows you to move from vague idea to realization making course corrections as you go.



# The Scrum Empirical Process



Foundation: \_\_\_\_\_



- |   |   |
|---|---|
| Answers questions on PBI's / Stories                  | A primary presenter at the sprint review          |
| Resolves technical impediments                        | Coaches the team                                  |
| Decides on process improvements for upcoming sprints  | Focuses on improving the definition of done       |
| Creates the potentially releasable increment          | Integrates, tests and deploys the product         |
| Determines how to approach the work and who does what | Responsible for quality                           |
| Resolves organizational and other impediments         | Facilitates Scrum events                          |
| Prioritizes product backlog                           | Cross functional and multi disciplinary           |
| Attends sprint retrospective                          | Most associated with the budget and ROI           |
| Protects the team                                     | Makes decisions on releasing the product          |
| Talks to stakeholders                                 | Attends sprint planning & review                  |
| Involved in backlog refinement                        | Decides how much work is accepted into the sprint |
| Designs and estimates implementation of PBIs          | Attends daily standup                             |
| Writes product backlog items                          | Responsible for hiring team members               |
| Expert on Scrum and guards the process                | Improves process and technical practices          |

# Sprint Planning Meeting

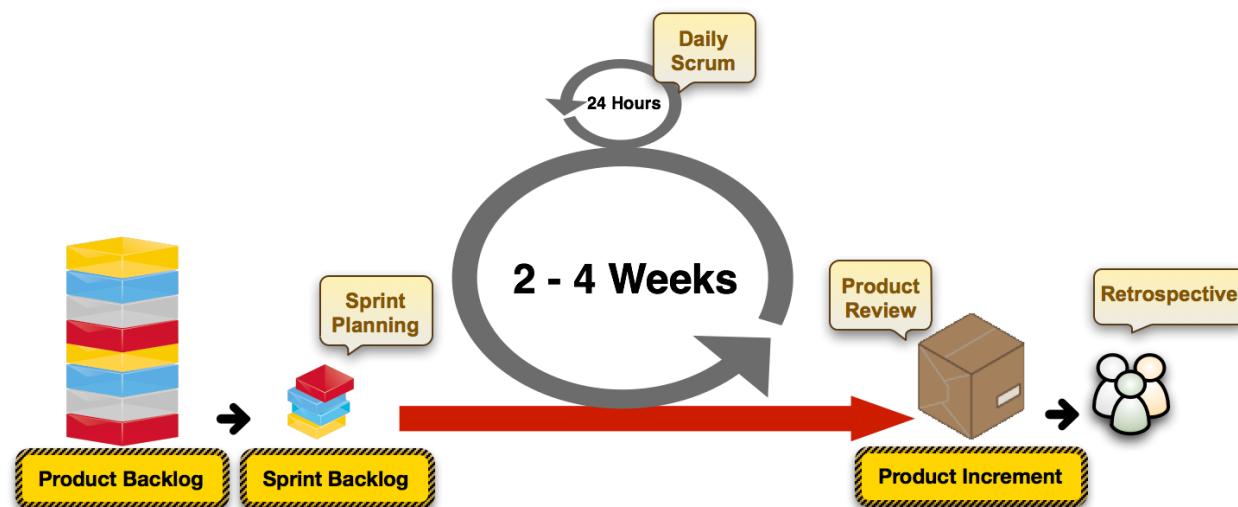
## WHAT (PO) + HOW (TEAM)

### What do we want? (PO)

- “What items are candidates for the Sprint?”
- Backlog items and acceptance criteria are finalized

### How will we do it? (Team)

- Stories broken down into tasks
- Stories are discussed and (re)estimated
- **PLAN TOGETHER**
- Definition of Done is reviewed
- Timeboxed (~1hr/week)



# Daily Scrum

- 15 minute meeting every day
- Same time, same place
- Three questions
  - What did I do since we last met?
  - What do I plan to do by our next meeting?
  - What impediments do I have?
- Focus is on Sprint commitments
- Not a status update! - **ALIGNMENT Meeting**



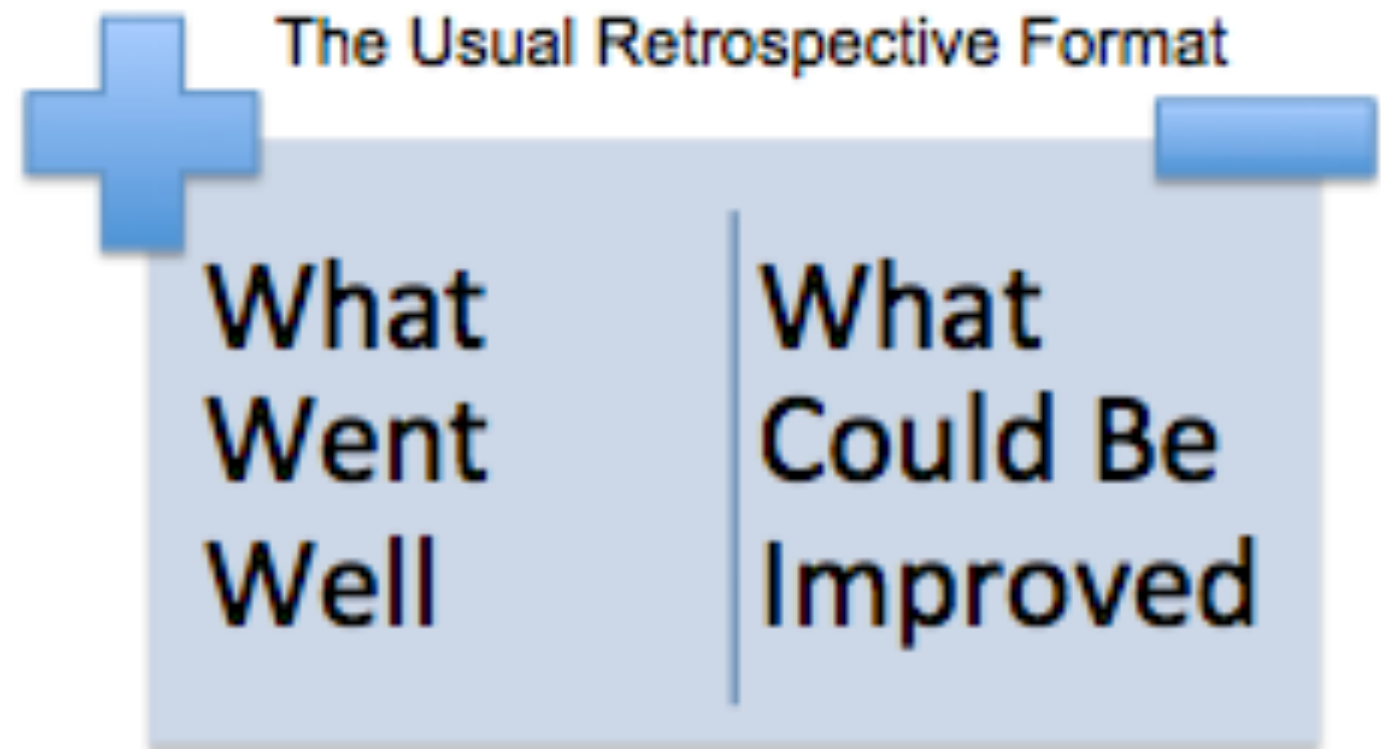


- **Product Review**

- Demonstrate functionality
- Show “Done” work
- Get sign-off from PO on A/C and DoD
- Gather feedback from stakeholders
- ~1 hr/week
- Celebrate!



- **Retrospective**



**ACTION ITEMS TO DO  
DURING NEXT SPRINT**  
(Review these during the  
next retrospective)

# Definition of Done

- Defines the work products that will be delivered with each item as it is ready for acceptance
- “Quality checklist”
  - *These must pass before we deploy our product to our customers*
  - *“Potentially releasable”*
  - *If multiple teams are working on one product, one DOD is shared across teams*
  - *Can include enterprise standards*
- Sprint cancelation?

## Definition of Done Brainstorm

User Story Defined	Code meets coding and naming standards	Automated acceptance tests run
Tasks Identified	Code review complete	Acceptance test run and pass 100%
Updated user guide	UI meets style standards	Integration test run
Updated online context help	Automated unit tests	Performance test run
Product Owner review and acceptance	Code check in to repository	Peer review
Zero defects	Unit tests pass 100%	Burndown chart ready
Build scripts updated	Code coverage run	Release build
Builds completing without error	Design-document updated	Functional testing



## Scrum Values

From <https://www.scrumalliance.org/about-scrum/values>

A team's success with Scrum depends on five values: **commitment, courage, focus, openness** and **respect**.

### The Five Scrum Values

The Scrum Guide lists five values that all Scrum teams share: commitment, courage, focus, openness, and respect.

#### How Does **Commitment** Allow Scrum Teams to Be Agile?

The Scrum value of commitment is essential for building an agile culture. Scrum teams work together as a unit. This means that Scrum and agile teams trust each other to follow through on what they say they are going to do. When team members aren't sure how work is going, they ask. Agile teams only agree to take on tasks they believe they can complete, so they are careful not to overcommit.

#### Great ScrumMasters Value Commitment.

Great ScrumMasters reinforce a team's commitment when they facilitate sprint planning, protect teams from mid-sprint changes, and deflect excessive pressure from product owners.

#### How Does **Courage** Allow Scrum Teams to Be Agile?

The Scrum value of courage is critical to an agile team's success. Scrum teams must feel safe enough to say no, to ask for help, and to try new things. Agile teams must be brave enough to question the status quo when it hampers their ability to succeed.

#### Great ScrumMasters Value Courage.

Great ScrumMasters help foster team courage by creating safety for team members to have difficult conversations--with one another, with the product owner, and with stakeholders. ScrumMasters are fearless about removing impediments that slow the team down. ScrumMasters also stand up to stakeholders to prevent changes or side projects during the sprint while also helping teams adapt when priorities shift between sprints.

#### How Does **FOCUS** Allow Scrum Teams to Be Agile?

The Scrum value of focus is one of the best skills Scrum teams can develop. Focus means that whatever Scrum teams start they finish--so agile teams are relentless about limiting the amount of work in process (limit WIP).

#### Great ScrumMasters Value Focus.

Great ScrumMasters encourage team focus by holding the team to their own definition of done, by encouraging full team participation at each daily scrum, and by ensuring that team members only present work that is complete at the sprint review.

#### How Does **Openness** Allow Scrum Teams to Be Agile?

Scrum teams consistently seek out new ideas and opportunities to learn. Agile teams are also honest when they need help.

#### Great ScrumMasters Value Openness.

Great ScrumMasters facilitate openness in daily scrums so the team is always aware of exactly how the sprint is going. ScrumMasters encourage openness in sprint reviews by ensuring that stakeholder feedback is constructive and that team members are able to hear it. ScrumMasters remind teams that learning about product shortcomings early is much less expensive and much more helpful than hearing about them late in the project, or worse, after the product is in customer's hands. In the same way, ScrumMasters foster an open environment in sprint retrospectives so that teams can grow and develop from sprint to sprint.

#### How Does **Respect** Allow Scrum Teams to Be Agile?

Scrum team members demonstrate respect to one another, to the product owner, to stakeholders, and to the ScrumMaster. Agile teams know that their strength lies in how well they collaborate, and that everyone has a distinct contribution to make toward completing the work of the sprint. They respect each other's ideas, give each other permission to have a bad day once in a while, and recognize each other's accomplishments.

#### Great ScrumMasters Value Respect.

Great ScrumMasters develop respect in their teams. They help teams listen to each other during daily scrums. They encourage teams to share their struggles and their successes. ScrumMasters also point out times of strong collaboration and facilitate conversations around new ideas.

# Product Owner: The Product Value Evangelist

**Forbes**insights

On a practical level, a Product Owner is responsible for defining the product vision, setting priorities to deliver the highest value and determining what project deliverables are important to a wide variety of stakeholders.

But in today's fast-paced, highly competitive world, a Product Owner must also be a visionary—someone who sits at the wheel of the car and determines which way to go.

Fluctuating needs and ongoing feedback can change the direction of a project. For this reason, a Product Owner must also be a good communicator. Not only does this involve the Product Owner explaining changing priorities and their varying impacts on different product teams, it also means listening to stakeholders and carefully managing their expectations.

Unfortunately, many organizations continue to treat Product Owners as glorified go-betweens—people who simply facilitate conversations between the team and management. But it's a mistake to treat a Product Owner as a proxy. Rather, a

Product Owner must be allowed to make critical decisions, no matter how unpopular.

Not everyone is built for the role of Product Owner. Courage is essential. After all, creating a vision for a product and determining a direction forward requires making choices that may encounter dissent, and persevering in the overall interest of maximizing value—both of the product, and for the customer.

Product Owners must also be highly self-disciplined. It can be tempting to try to control the work of others. Experienced Product Owners know not to try to manage the Scrum Team's activities.

<https://www.forbes.com/sites/insights-scrumalliance/#1c90c9b81598>

From “The Anatomy Of An Excellent Scrum Team” article John Miller, CST, helped Forbes to write.

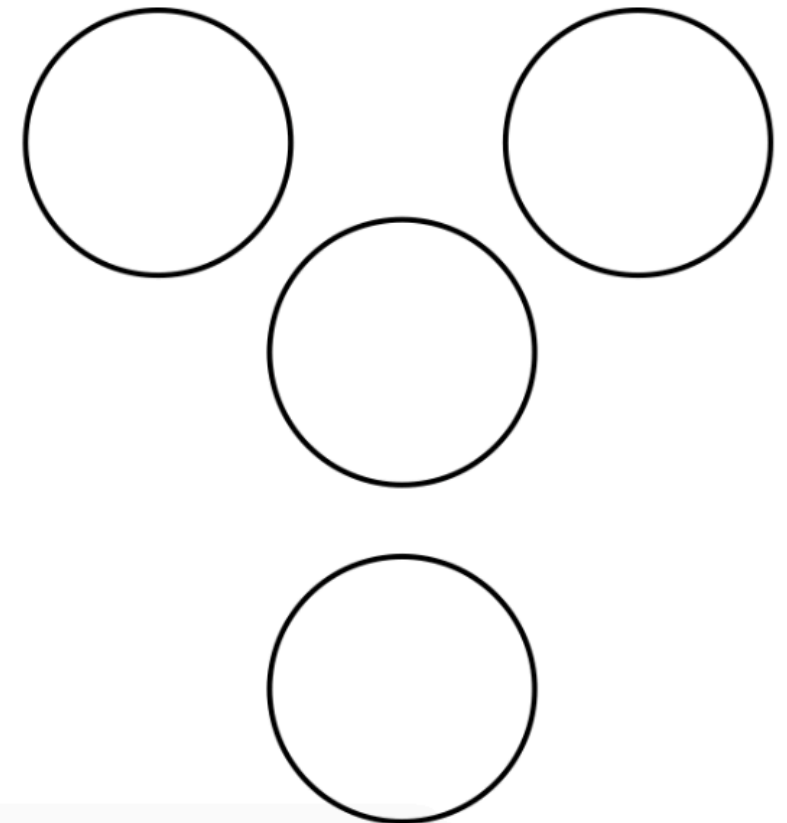


# Product Owner - “Value Driver”

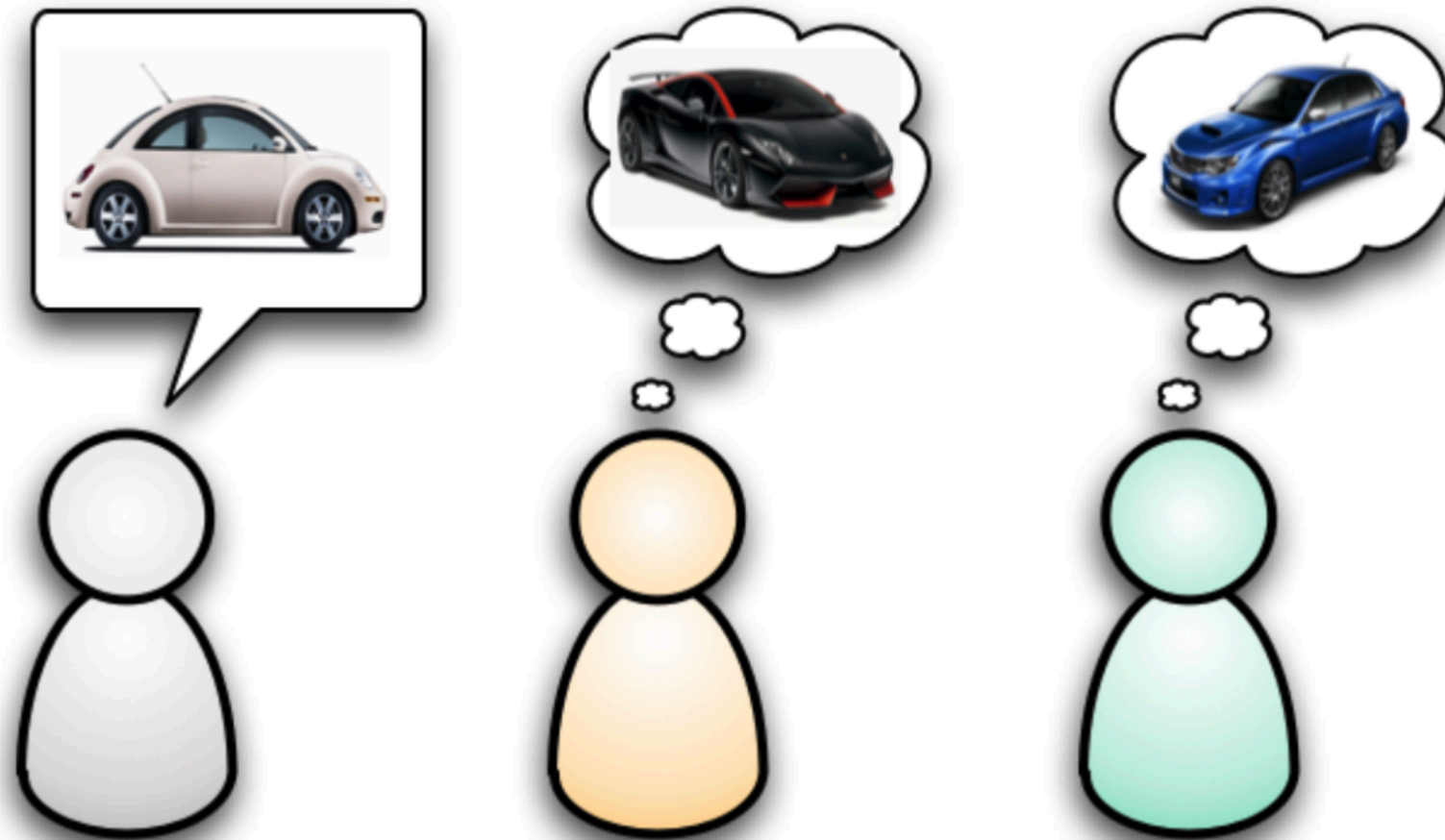
- Represents (or is) the user or customer
- One voice to team
- Drives product success - Understands constraints and ROI
- Create the Product Vision - The “end goal”
- **Creates and maintains the Product Backlog - Has authority over priorities**
- Participates in Sprint Meetings
- Key characteristics: \_\_\_\_\_

# Product Owner - Additional Duties

- Makes scope vs. schedule tradeoff decisions
- Manages all stakeholders and their diverse interests
- Accepts or rejects work of the team (sign-off)
- **Grooms the backlog - Helps the team understand next pieces of work, early introduction to requirements**
- Works with a one Scrum team...

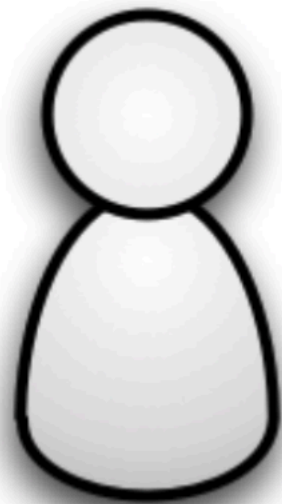


# Product Ownership is Creating a Shared Understanding



**“I want a SPORTS CAR.”**

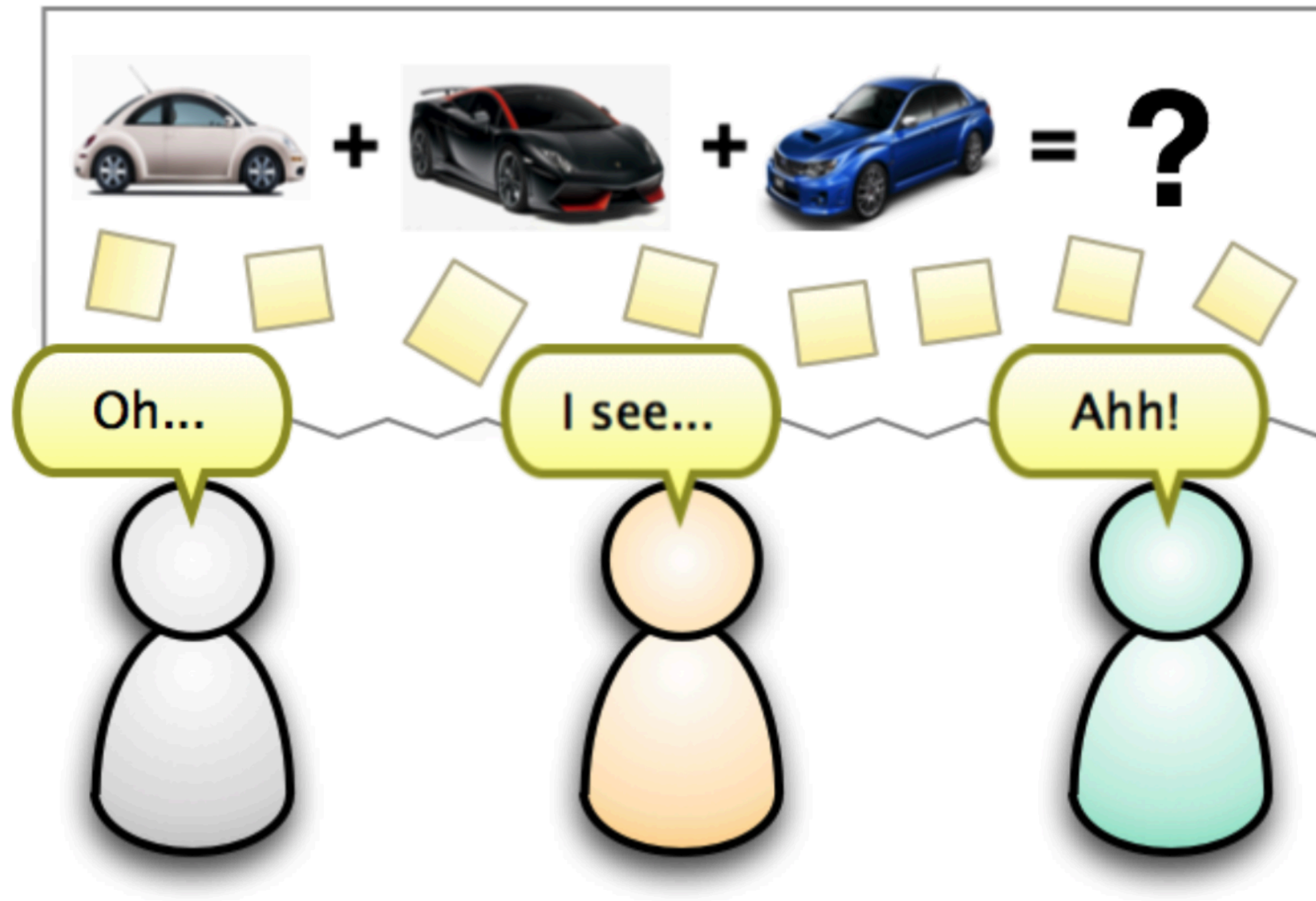
# Visual Collaboration + Discussion



Discussion + drawing simple models  
help us detect differences

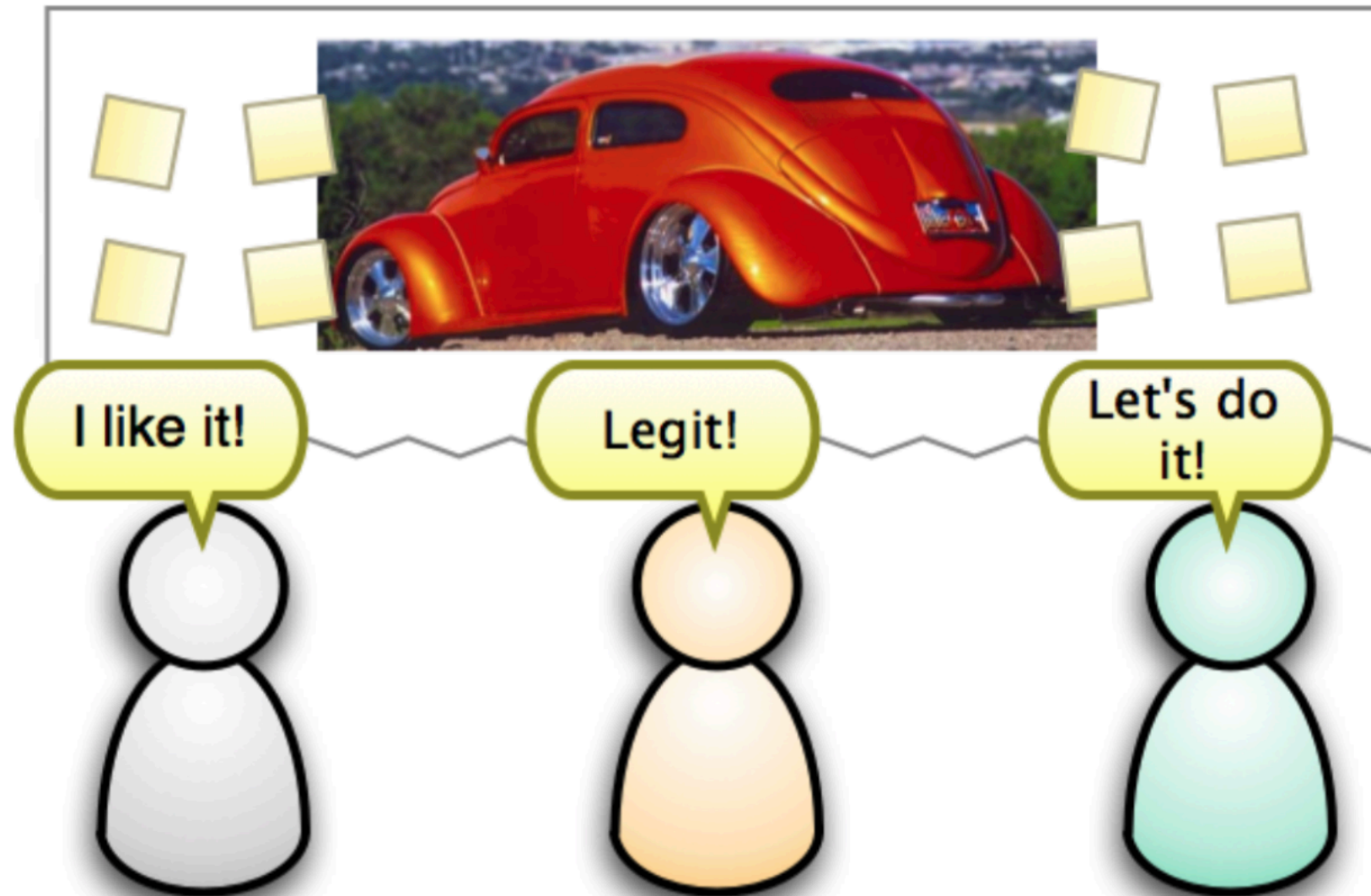


# Coming to a Shared Understanding



**“I want a SPORTS CAR.”**

# When We Say the Same Thing... We Mean It!



“I’m glad we agree now.”



# Sharing Documentation $\neq$ Shared Understanding



# The Prime Directive of Product Management

- To build the **minimum** you can.
- To build the **least amount** you can.
- To **smallest amount** you can.
- To **do the least** you can.
- To have **everyone understand** (aligned) around what you want built.

*“The best way to predict your future is to create it.”*

*- Abraham Lincoln*



# Spaghetti Dinner Recipe



## Your Amazing Recipe

1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_
4. \_\_\_\_\_
5. \_\_\_\_\_
6. \_\_\_\_\_
7. \_\_\_\_\_
8. \_\_\_\_\_
9. \_\_\_\_\_
10. \_\_\_\_\_

# Start With Who

*How would these personas want their spaghetti dinner made? What top 3 'features' would each have?*

## Little Kid



- Feature 1: \_\_\_\_\_
- Feature 2: \_\_\_\_\_
- Feature 3: \_\_\_\_\_

## Paleo



- Feature 1: \_\_\_\_\_
- Feature 2: \_\_\_\_\_
- Feature 3: \_\_\_\_\_

## Vegetarian



- Feature 1: \_\_\_\_\_
- Feature 2: \_\_\_\_\_
- Feature 3: \_\_\_\_\_

## Gordon Ramsey



- Feature 1: \_\_\_\_\_
- Feature 2: \_\_\_\_\_
- Feature 3: \_\_\_\_\_

*1. How are these different than the initial recipe your group made?*

---

---

*2. What is the lesson here?*

---

---

# Simulation - Product Backlog to Delivery

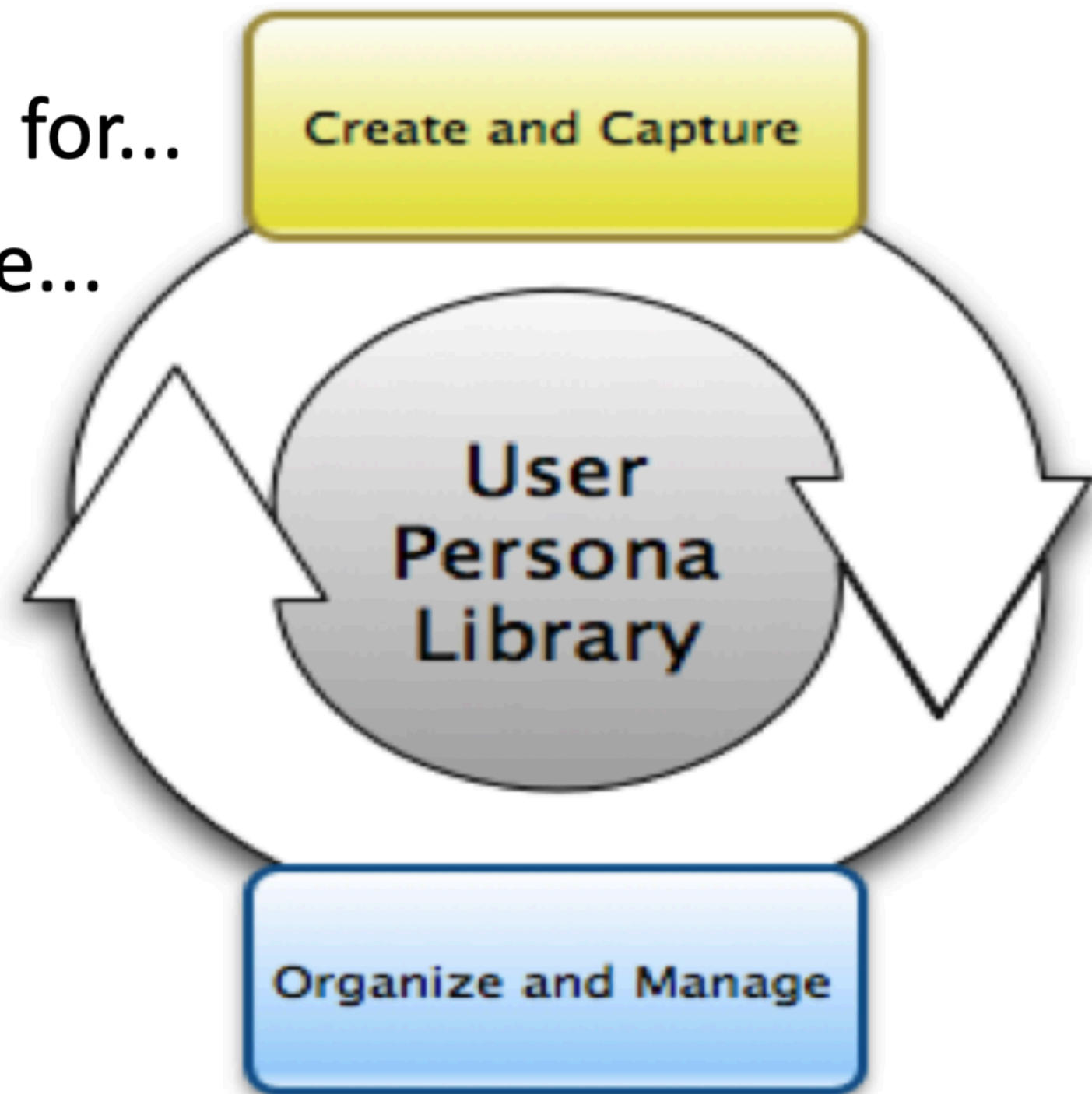
## **CHOOSE A PRODUCT:**

- 1.iPhone mobile app
- 2.New product line/idea
- 3.Enhancing an existing product
- 4.Internal product release
- 5.Infrastructure support
- 6.Operational support



# Value & Features Begin with Users (Personas)!

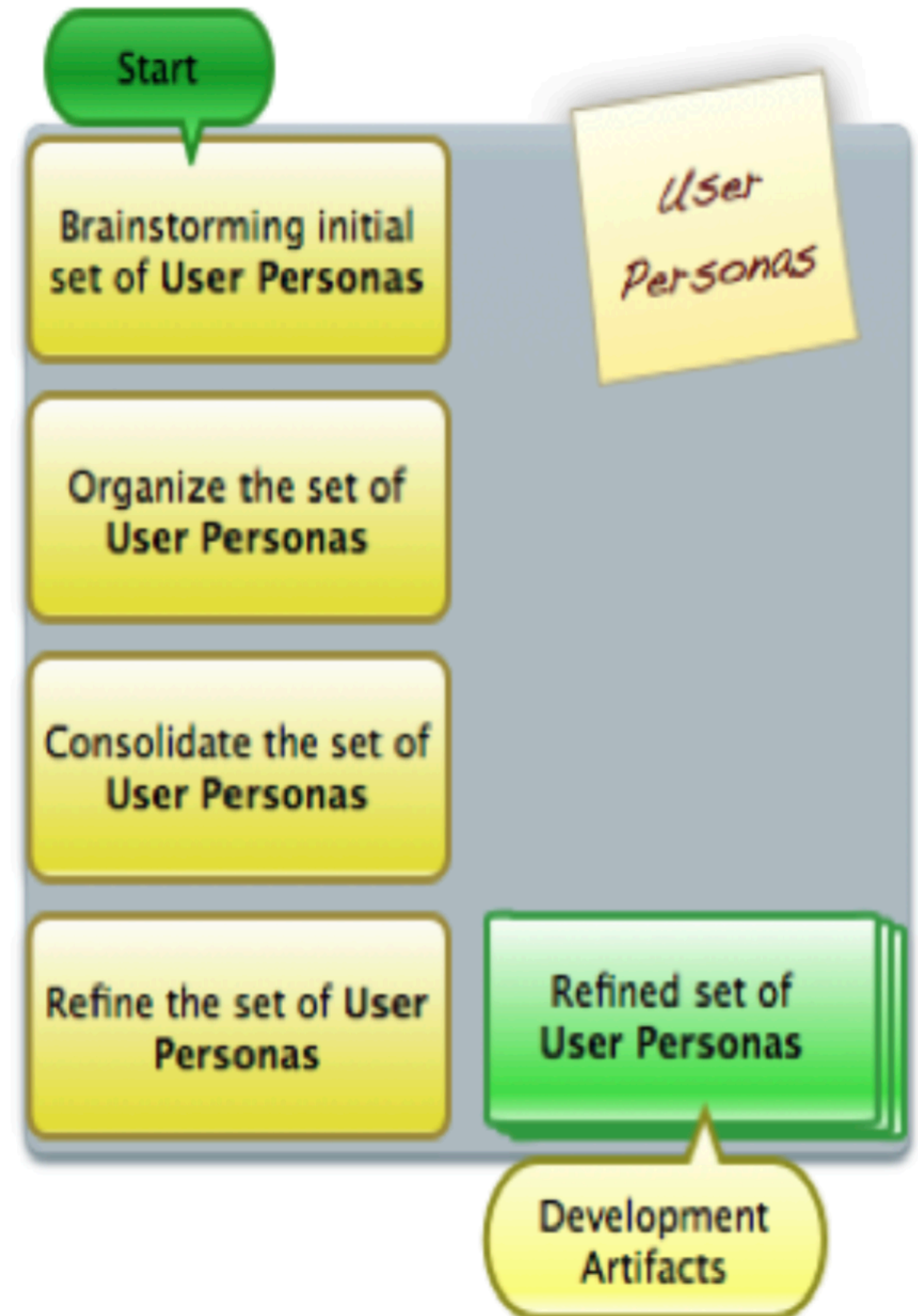
- Think about all users...
- What they use the system for...
- How they use the software...
- Background / Values...





# Exercise - User Persona Brain Storming

- Bring customers, developers, anyone who understands a product's intended users
- Write role names on stickies
  - As fast as possible with no judgement
    - No turns / SILENTLY
  - Discuss what is meant by them
    - Organize, refine, eliminate
    - Define each Persona by one or two permissions



# The First Filters for Value

1. \_\_\_\_\_

2. \_\_\_\_\_

3. \_\_\_\_\_

4. \_\_\_\_\_

## 4 Product Owner Questions

### 4 Questions a PO asks about value

1. Is it needed?
2. Is it usable?
3. Is it buildable?
4. Is it supportable?

## 4 Questions for a PO asks of themselves

1. Do I have deep customer knowledge? If not, how do I get it and fast!
2. Can I collect and interpret the right data?
3. Do I have knowledge of **my/our** business?
4. Do I have market/industry knowledge?



# The Elements of Value Pyramid

Products and services deliver fundamental elements of value that address four kinds of needs: functional, emotional, life changing, and social impact. In general, the more elements provided, the greater customers' loyalty and the higher the company's sustained revenue growth.

## SOCIAL IMPACT



Self-transcendence

## LIFE CHANGING



Provides hope



Self-actualization



Motivation



Heirloom



Affiliation/  
belonging

## EMOTIONAL



Reduces anxiety



Rewards me



Nostalgia



Design/  
aesthetics



Badge  
value



Wellness



Therapeutic  
value



Fun/  
entertainment



Attractiveness



Provides  
access

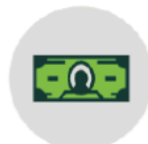
## FUNCTIONAL



Saves  
time



Simplifies



Makes  
money



Reduces  
risk



Organizes



Integrates



Connects



Reduces  
effort



Avoids  
hassles



Reduces  
cost



Quality



Variety



Sensory  
appeal



Informs

# The Elevator Statement

- **FOR** (target customer)
- **WHO** (statement of the need)
- **THE** (product name) is a (product category)
- **THAT** (product key benefit, compelling reason to buy).
- **UNLIKE** (primary competitive alternative),
- **OUR PRODUCT** (final statement of primary differentiation)

# Example Product Elevator Statement

**For** dentists and their assistants  
**Who** need to efficiently schedule appointments  
**The *Dental Clinic 2.0* is** desktop and web-based  
appointment scheduling software  
**That** supports office and remote access.  
**Unlike** all competitors  
*Dental Clinic 2.0 is easy to use.*

# Vision Examples & Characteristics

## Ford, 1905

We will build a motor car for the great multitudes...It will be so low in price that no man making a good salary will be unable to own one and enjoy with his family the blessing of hours of pleasure in God's great open spaces.... When we are through, everyone will be able to afford one.... the horse will have disappeared from our highways and the automobile will be taken for granted.

### Great Visions Are:

1. Future Focused
2. Clear
3. Relevant
4. Challenging
5. Unique
6. Vivid
7. Inspiring

## Sony, 1950s

We will create products that become pervasive around the world.... We will be the first Japanese company to go into the American market and distribute directly.... We will succeed with innovations like the transistor radio that American companies have failed at.... Fifty years from now, our brand-name will be as well-known as any on Earth ... and will signify innovation and quality that rivals the most innovative companies anywhere.... "Made in Japan" will mean something fine, not shoddy.

## Agile For All Online 2018

In 2025, when someone needs to improve their capability in an Agile organization, they choose online training from Agile For All to get the skills they need right when they need them, without any sense that they're settling by not waiting for an in-person class.

This online training is so engaging that people look forward to a new release of content the same way they anticipate a new series on Netflix or their gaming platform.

It has enabled people in every socioeconomic status, in any location in the world, to be more prosperous and thrive by discovering how to do meaningful work in a state of flow.

Agile For All team members continue to have a deep impact on clients but no longer spend days traveling away from their families and communities.



## Vision Practice

### Tips:

- 1. Keep it focused on what customers will do, not on what products/features you'll build*
- 2. Speak in the present tense. In 20xx, customers do this (not customer will do this)*
- 3. Talk about the why.*
- 4. Make it concrete (no 'yogababble' ).*

### Draft 1:

### Draft 2:

### Draft 3:

---

## **Review your new mindset**

Find one or two other people that aren't sitting at your table and review the material covered so far.

**List 3 things you have learned so far that seem crucial to you:**

**List 2 things you are going to do differently at work once this workshop is completed:**

**What was your biggest surprise so far (and do not pick Multitasking is horrible!)?**

# From Concept to Product Backlog

- Gather raw data from users (external / internal)
  - **Interviews, focus groups, observation**, etc.
  - Be prepared to let the user run the product
  - Listen for latent needs, watch for **nonverbal** clues
- Formulate raw data into user needs
  - Convert user statements into description of interpreted user need or value
  - Focus on WHAT, not HOW
  - Avoid adding extra detail through interpretation or assumptions

# Ground Rules for Interviewing



It is an art to conduct good interviews that provide relevant insights for value proposition design. Make sure you focus on unearthing what matters to (potential) customers rather than trying to pitch them solutions. Follow these rules to conduct great interviews.

## Rule 1

### Adopt a beginner's mind

Listen with a "fresh pair of ears" and avoid interpretation. Explore unexpected jobs, pains, and gains in particular.

## Rule 2

### Listen more than you talk

Your goal is to listen and learn, not to inform, impress, or convince your customer of anything. Avoid wasting time talking about your own beliefs, because it's at the expense of learning about your customer.

## Rule 3

### Get facts, not opinions

Don't ask, "Would you...?"

Ask, "When is the last time you have...?"

## Rule 4

### Ask "why" to get real motivations

Ask, "Why do you need to do...?"

Ask, "Why is \_\_\_ important to you?"

Ask, "Why is \_\_\_ such a pain?"

## Rule 5

### The goal of customer insight interviews is not selling (even if a sale is involved); it's about learning

Don't ask, "Would you buy our solution?"

Ask "what are your decision criteria when you make a purchase of...?"

## Rule 6

### Don't mention solutions (i.e., your prototype value proposition) too early

Don't explain, "Our solution does..."

Ask, "What are the most important things you are struggling with?"

## Rule 7

### Follow up

Get permission to keep your interviewee's contact information to come back for more questions and answers or testing prototypes.

## Rule 8

### Always open doors at the end

Ask, "Who else should I talk to?"



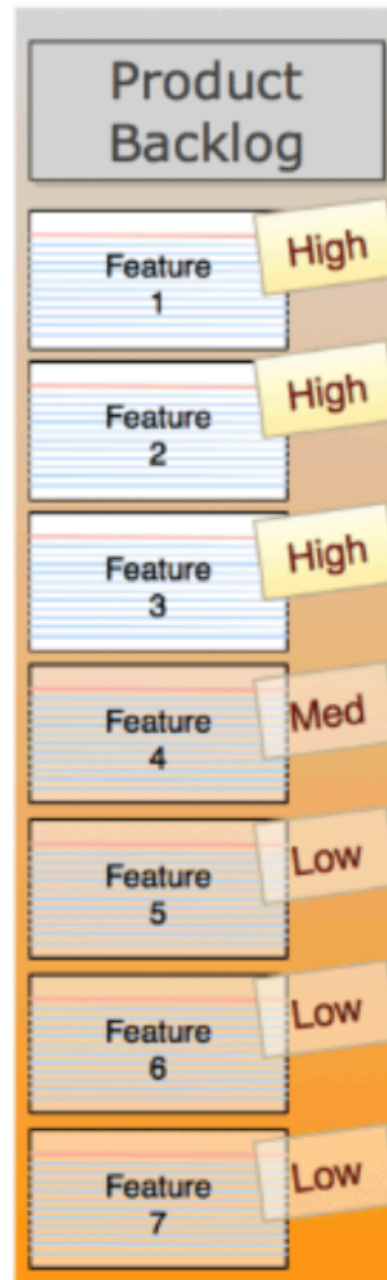
# Product Backlog - It Starts with YOU

- Highest priority items first at the top (value)
- Items can split anytime
- Items can be removed anytime
- Items can be added anytime
- Items can be re-prioritized anytime

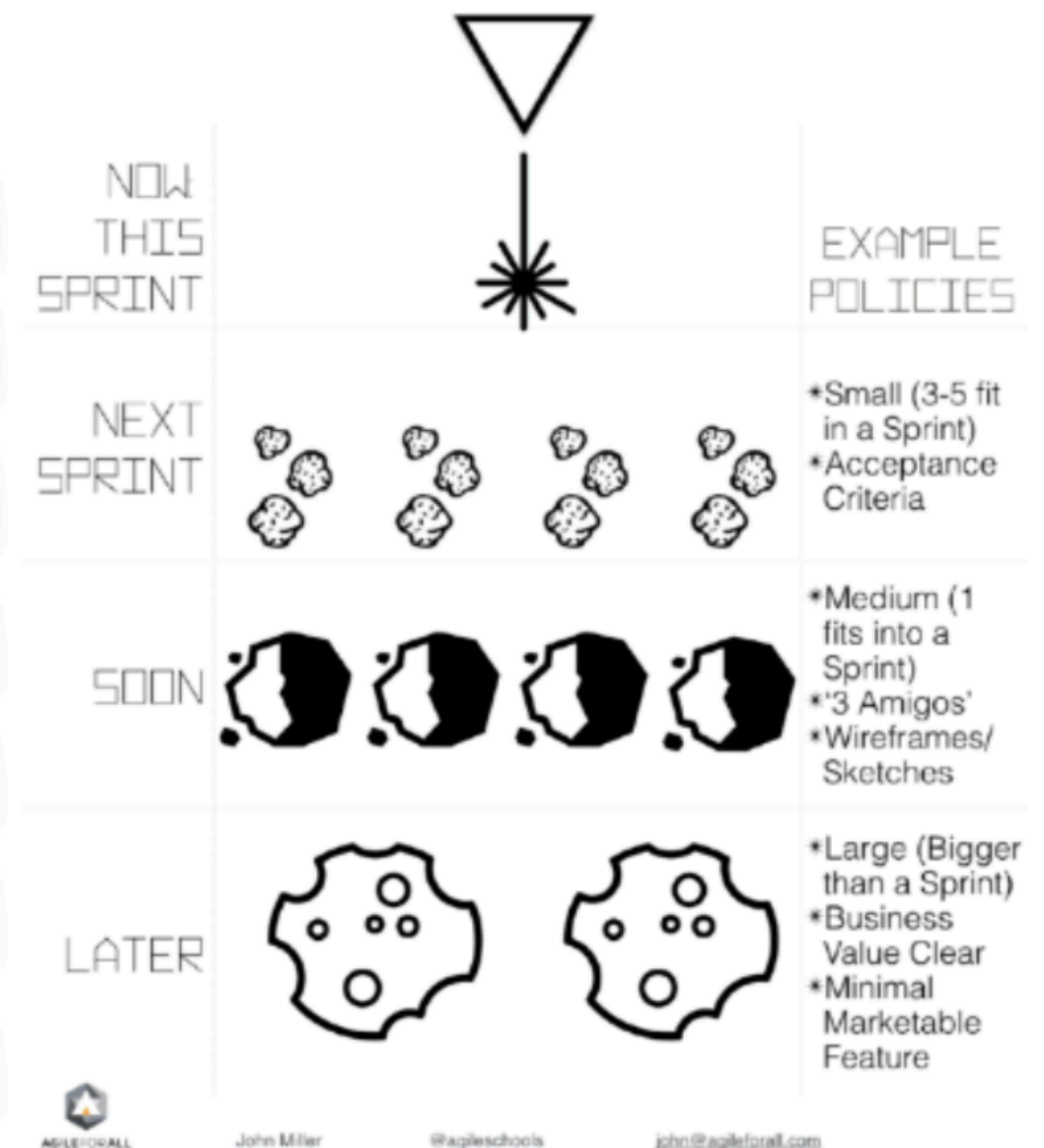
Sprint Backlog	Stories	Tasks	Tasks In Progress	Completed
	#1			
	#2			
	#3			
	#4			
	#5			

# Product Backlog Refinement

- Items can split, removed, changed, added, re-prioritized anytime
- ▢ A Product Backlog Item (Story) is ready if it is understandable by both the **Product Owner** and the team - Work together!
- ▢ Product Backlog Items cannot be groomed appropriately if the delivery team(s) haven't helped



## BACKLOG ASTEROIDS



# 3 C's of User Stories

- **CARD** - Small... so that...
- **CONVERSATION** - Collaborative exploration
- **CONFIRMATION** - Acceptance Criteria

**AS A** <type of user>  
**I WANT TO** <goal>  
**SO THAT** <value>

Acceptance Criteria:

# 3C's + I.N.V.E.S.T. of User Stories

- **CARD**: Token representing the requirement. Notes are written on it, reflecting priority and cost
  - **CONVERSATION**: Requirement is communicated from customer to team to refine details (often supplemented with documents)
  - **CONFIRMATION**: Acceptance tests make it possible to understand what needs to be done to complete the requirement
- **Independent** - schedule and implement in any order
  - **Negotiable** - able to change
  - **Valuable** - must be valuable to customer
  - **Estimable** - to help customer rank based on value and cost
  - **Small** - consumable by a team for implementation
  - **Testable** - shows story is clear and defines done state



# Acceptance Criteria

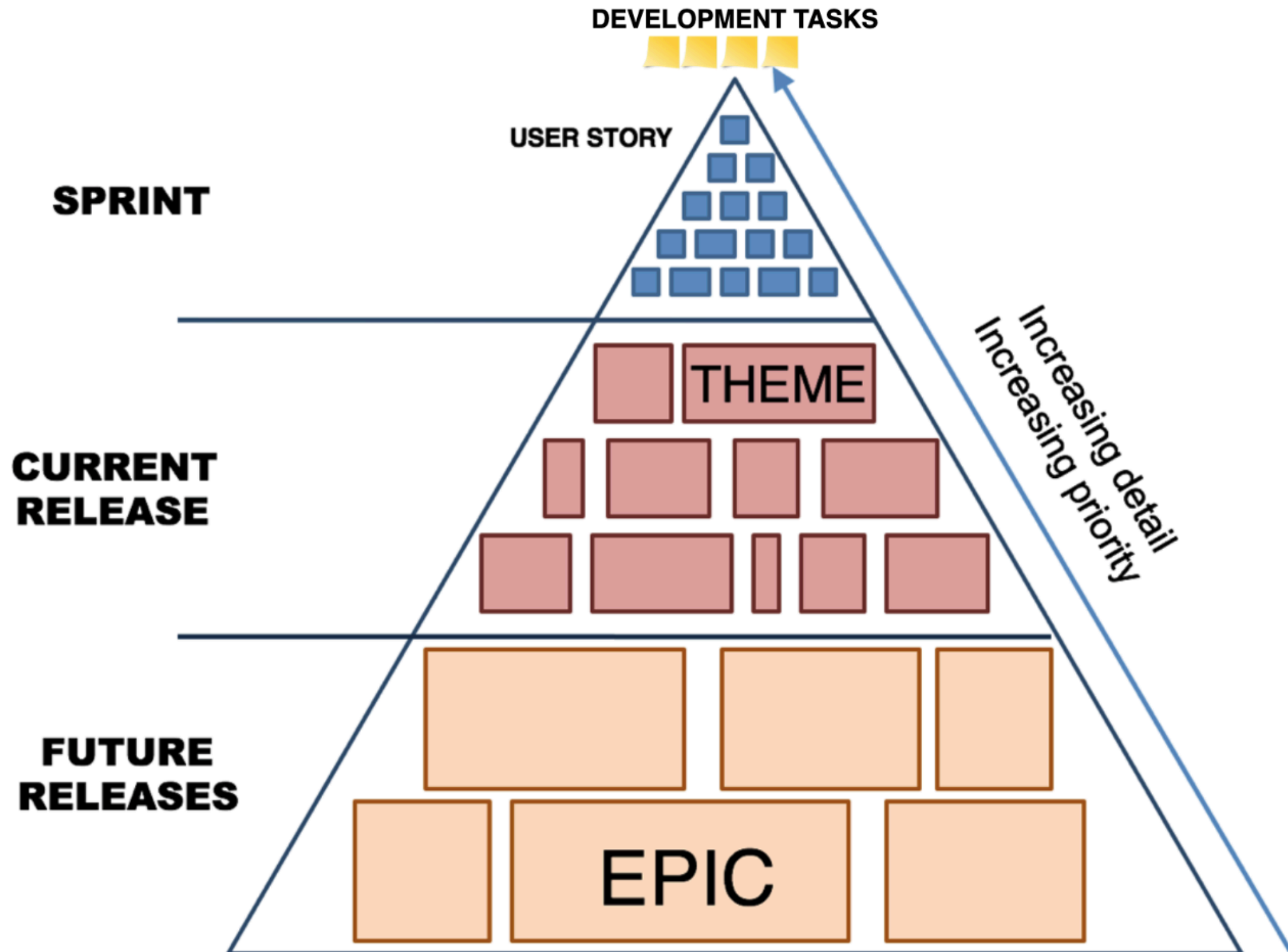
- A/C = “DONE” means for a User Story
- Product Owner takes first pass at this before planning, then **negotiates** with the team

**AS A** user, I can cancel a reservation...

AC:

1. Verify that a premium member can cancel the same day without a fee.
2. Verify that a non-premium member is charged 10% for a same-day cancelation
3. Verify that an email confirmation is sent.
4. Verify that the hotel is notified of any cancelation

# EPIC (Large) - User Stories - Split Them!



**THEME -> EPIC -> USER STORY**

# HOW TO SPLIT A USER STORY

## 1 PREPARE THE INPUT STORY



## WORKFLOW STEPS

Can you split the story so you do the beginning and end of the workflow first and enhance with stories from the middle of the workflow?

Can you take a thin slice through the workflow first and enhance it with more stories later?

## DEFER PERFORMANCE

Could you split the story to just make it work first and then enhance it to satisfy the non-functional requirement?

Does the story get much of its complexity from satisfying non-functional requirements like performance?

## SIMPLE/COMPLEX

Could you split the story to do that simple core first and enhance it with later stories?

Does the story have a simple core that provides most of the value and/or learning?

Could you group the later stories and defer the decision about which story comes first?

## MAJOR EFFORT

When you apply the obvious split, is whichever story you do first the most difficult?

## 2 APPLY THE SPLITTING PATTERNS

start here

## OPERATIONS

Can you split the operations into separate stories?

Does the story include multiple operations? (e.g. is it about "managing" or "configuring" something?)

## BUSINESS RULE VARIATIONS

Can you split the story so you do a subset of the rules first and enhance with additional rules later?

Does the story have a variety of business rules? (e.g. is there a domain term in the story like "flexible dates" that suggests several variations?)

## VARIATIONS IN DATA

Can you split the story to process one kind of data first and enhance with the other kinds later?

Does the story do the same thing to different kinds of data?

Does the story have a complex interface?

Is there a simple version you could do first?

## INTERFACE VARIATIONS

Can you split the story to handle data from one interface first and enhance with the others later?

Does the story get the same kind of data via multiple interfaces?

## BREAK OUT A SPIKE

Are you still baffled about how to split the story?

Can you find a small piece you understand well enough to start?

Write that story first, build it, and start again at the top of this process.

Can you define the 1-3 questions most holding you back?

Write a spike with those questions, do the minimum to answer them, and start again at the top of this process

Take a break and try again.

## 3 EVALUATE THE SPLIT

Are the new stories roughly equal in size?

**YES** → Is each story about 1/4 to 1/2 of your velocity?

**NO** → Try another pattern on the original story or the larger post-split stories.

Do each of the stories satisfy INVEST?

Are there stories you can deprioritize or delete?

Try another pattern.

Try another pattern. You probably have waste in each of your stories.

Is there an obvious story to start with that gets you early value, learning, risk mitigation, etc.?

Try another pattern to see if you can get this.

You're done, though you could try another pattern to see if it works better.

\* INVEST - Stories should be:  
Independent  
Negotiable  
Valuable  
Estimable  
Small  
Testable



AGILEFORALL  
www.agileforall.com

Visit <http://www.richardlawrence.info/splitting-user-stories/> for more info on the story splitting patterns  
Copyright © 2011-2018 Agile For All. All rights reserved.

Last updated 2/21/2018



# Story Splitting Cheat Sheet

---

## The INVEST Model

Stories should be: Independent, Negotiable, Valuable, Estimable, Small, and Testable.

## Patterns for Splitting Stories

### Workflow Steps

As a content manager, I can publish a news story to the corporate website.

...I can publish a news story directly to the corporate website.  
...I can publish a news story with editor review.  
...I can publish a news story with legal review.

### Business Rule Variations

As a user, I can search for flights with flexible dates.

...as "n days between x and y."  
...as "a weekend in December."  
...as " $\pm$  n days of x and y."

### Major Effort

As a user, I can pay for my flight with VISA, MasterCard, Diners Club, or American Express.

...I can pay with one credit card type (of VISA, MC, DC, AMEX).  
...I can pay with all four credit card types (VISA, MC, DC, AMEX).

### Simple/Complex

As a user, I can search for flights between two destinations.

...specifying a max number of stops.  
...including nearby airports.  
...using flexible dates.  
...etc.

### Variations in Data

As a content manager, I can create news stories.

...in English.  
...in Japanese.  
...in Arabic.  
...etc.

### Data Entry Methods

As a user, I can search for flights between two destinations.

...using simple date input.  
...with a fancy calendar UI.

### Defer Performance

As a user, I can search for flights between two destinations.

...(slow - just get it done, show a "searching" animation).  
...(in under 5 seconds).

### Operations (e.g. CRUD)

As a user, I can manage my account.

...I can sign up for an account.  
...I can edit my account settings.  
...I can cancel my account.

### Break Out a Spike

As a user, I can pay by credit card.

Investigate credit card processing.  
Implement credit card processing (as one or more stories).

# User Story Writing - Scrum Simulation

- Write stories that represent the items (20 stories)
- Divide and conquer on different ideas/themes
- All - place the stories on Backlog
- 20 minutes

Stories	Tasks	Tasks In Progress	Completed
<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>			

# Exercise - Splitting + Acceptance Criteria

- Add acceptance criteria to your stories + **SPLIT THEM!**
- Remember to **verify testability**
- **See if you can break user stories down from AC**
- Ask the questions:
  - “Are there things we can make smaller?”
  - “Are there opportunities to get faster feedback from our users?”
  - “Can we test it?”

**AS A <type of user>  
I WANT TO <goal>  
SO THAT <value>**

**Acceptance Criteria:**

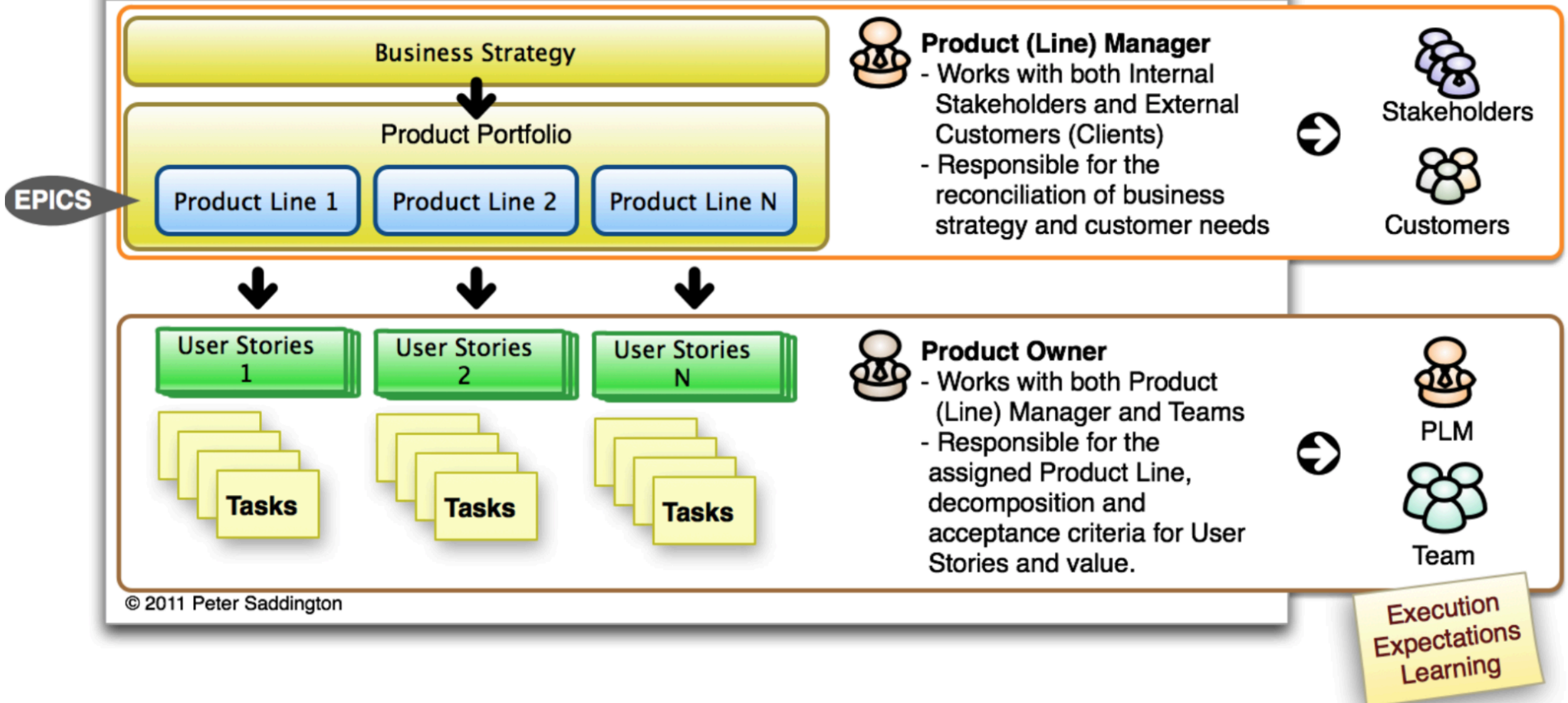
# Basics on When to Plan (Minimum)

Planning Level	Frequency	Who	Focus
<b>Product</b>	1-2 times per year	Product owner and executives	Product evolution over time
<b>Release</b>	3-4 times per year	Product owner and team	Tradeoffs between features and delivery date
<b>Sprint</b>	Every 2-4 weeks	Product owner and team	What features can be delivered within the sprint
<b>Daily</b>	Every day	Team	How to complete committed features



# Scaling the Product Owner

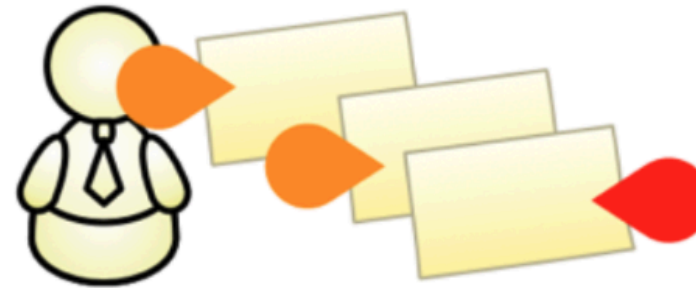
## Product Ownership - Portfolio Epics and Product (Line) Managers and Product Owner Roles



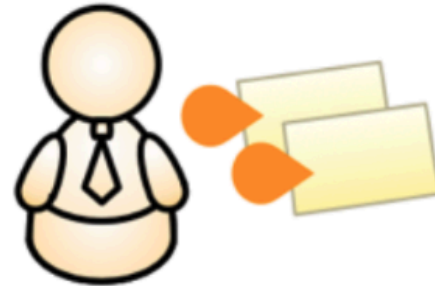
# Sharing the Product Backlog

## Scaling Product Ownership with Shared Backlogs and Alignment Meetings

**Chief Product Owner**



**Shared Backlog**



**Product (Line) Owners  
- Attend Scrum of Scrums**



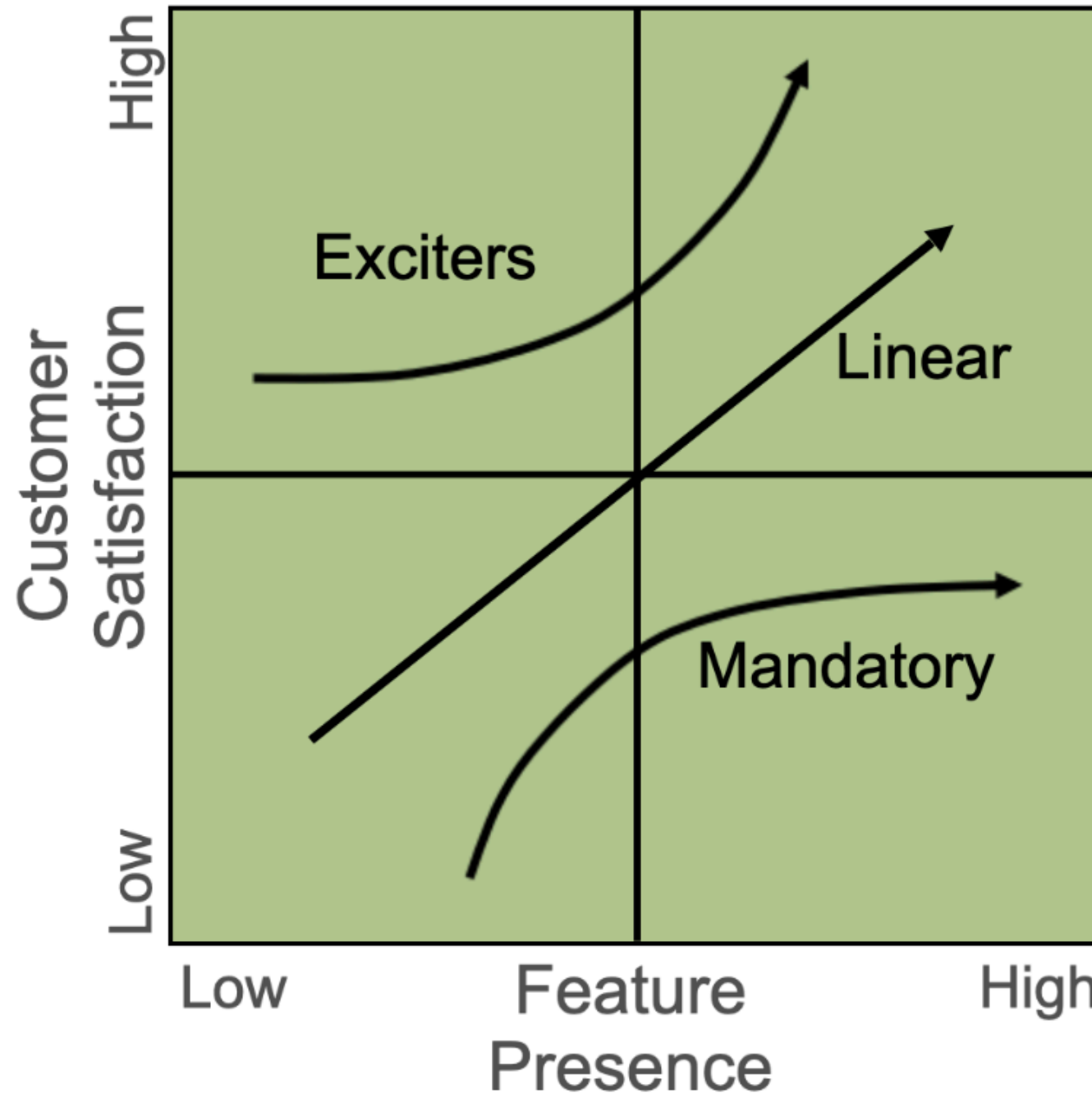
**Product Owners  
- Attend Daily Scrums**



# Prioritization - Kano Analysis

- Three types of features:
  - **Mandatory / Baseline** - Must be present in order for users to be satisfied
  - **Linear** - The more of it, the better
  - **Exciters / Delighters** - Features a user doesn't know she wants, until she sees it

# Kano - Impact on Customer Satisfaction





# Kano - Survey, Survey, Survey!

- To assess whether a feature is baseline, linear, or exciting we can:
  - Sometimes guess...
  - Or survey a small set of users (20-40)
- We ask two questions...
  - A functional question:
    - How do you feel if a feature is present?
  - And a dysfunctional question
    - How do you feel if that feature is absent?

# Kano - Functional and Dysfunctional Forms

Functional  
form of  
question

If your hotel room  
includes a free  
bottle of water, how  
do you feel?

I like it that way.	X
I expect it to be that way.	
I am neutral.	
I can live with it that way.	
I dislike it that way.	

Dysfunctional  
form of  
question

If your hotel room  
does not include a  
free bottle of water,  
how do you feel?

I like it that way.	
I expect it to be that way.	X
I am neutral.	
I can live with it that way.	
I dislike it that way.	

# Kano - Categorizing an Answer Pair

		Dysfunctional Question				
		Like	Expect	Neutral	Live with	Dislike
Functional Question	Like	Q	E	E	E	L
	Expect	R	I	I	I	M
	Neutral	R	I	I	I	M
	Live with	R	I	I	I	M
	Dislike	R	R	R	R	Q

M Mandatory  
 L Linear  
 E Exciter  
 Q Questionable  
 R Reverse  
 I Indifferent

# Kano - Aggregating Results

Theme	Exciter	Linear	Mandatory	Indifferent	Reverse	Questionable
Apply formatting themes	3	11	31	1	3	2
Automate report execution	4	22	20	4	1	0
Export reports to PowerPoint	21	9	14	5	1	1

**Grab the highest!**



# Kano - What to Include

- All of the mandatory (baseline) features
  - By definition, these must be present
- Some amount of linear features
- But leaving room for at least a few excitors
- Also experiment and leave margin for tests

# Theme Screening

- Identify around 5-9 selection criteria for what is important in the next release
- Select a baseline theme
  - Likely to be included in the next release
  - Understood by most team members
  - Assess each candidate theme relative to the baseline theme

# Example Theme Screening

Selection Criteria		Themes						
		Add a Graph	Add Export	Add Twitter API	Baseline Theme	Build Version 2	Add Custom UI	Mobile Access
Value to Current Customers		+	+	-	0	-	+	0
Increase Sales		+	-	0	0	0	0	0
Decrease Cost		+	0	0	0	+	-	+
Makes us Competitive in Market		0	0	0	0	+	0	+
+ = better than								
0 = same as								
- = worse than								
Net score		3	0	-1	0	1	0	2
Rank		1	4	5	4	3	4	2
Continue?		Y	N	N	Y	Y	N	Y

# Product Owner Expert Opinion

- Focus needs to be on delivering value to the customer
  - Surveys, collaboration, communication
- Product Owner empowerment allows you ability to be a trump card
- Always consider these four business factors...
  - Delivery of new capabilities
  - Development of new knowledge
  - Mitigation of risk
  - Changes in relative cost





# Simplified Kano + Expert Opinion

- Review your stories and group them in features:

- Mandatory
- Linear
- Exciter

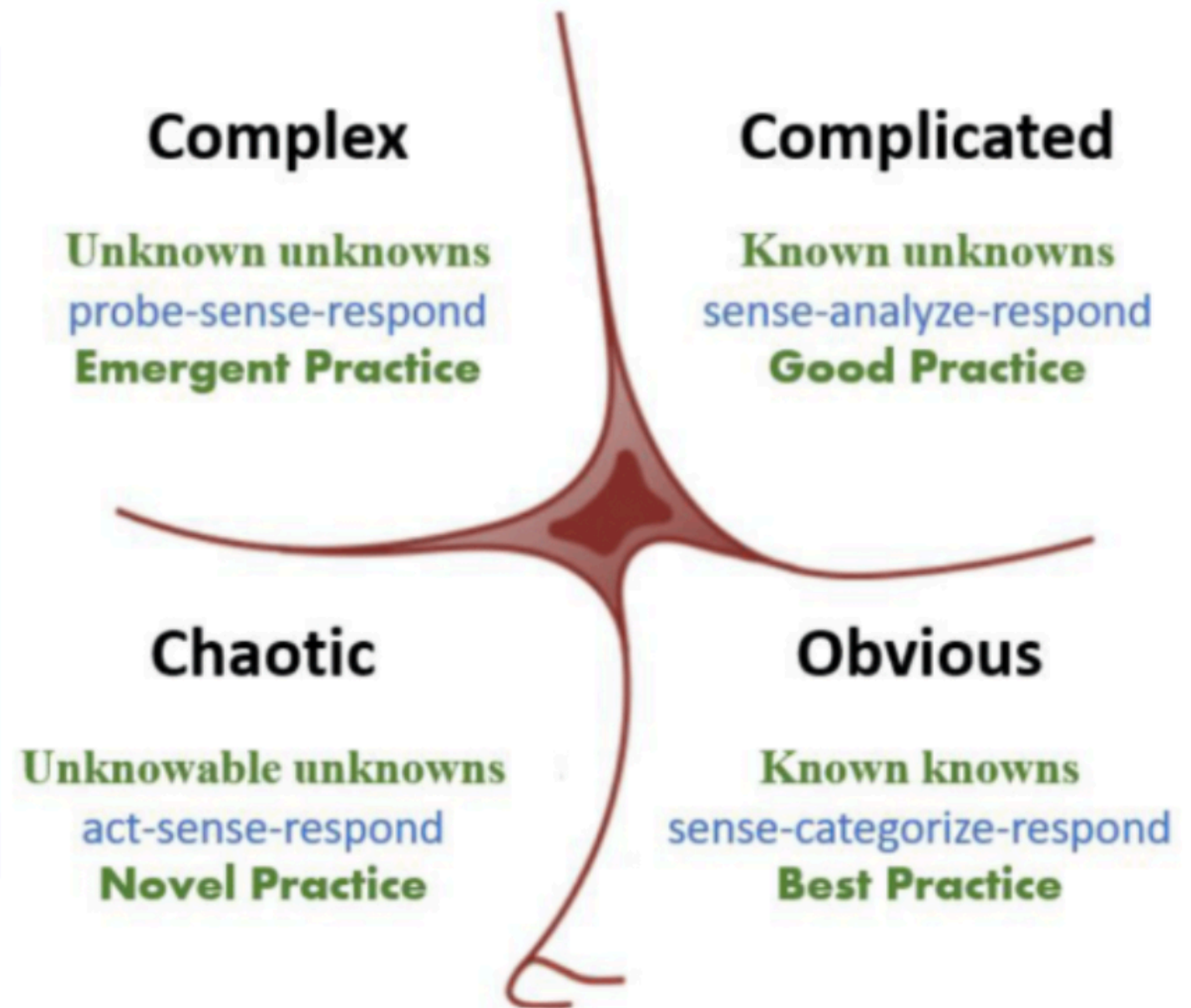
	Story A	Story B	Story C	Story N
Mandatory	III	IIIIIIII	I	
Linear	IIIIIIII	II	I	
Exciter	III	IIII	IIIIIIIIII	

- Now our break them out into different factors:

- Delivery of new capabilities ✓
- Development of new knowledge !
- Mitigation of risk ✗
- Changes in relative cost △



# The Current World & Cynefin (kun-EV-in)



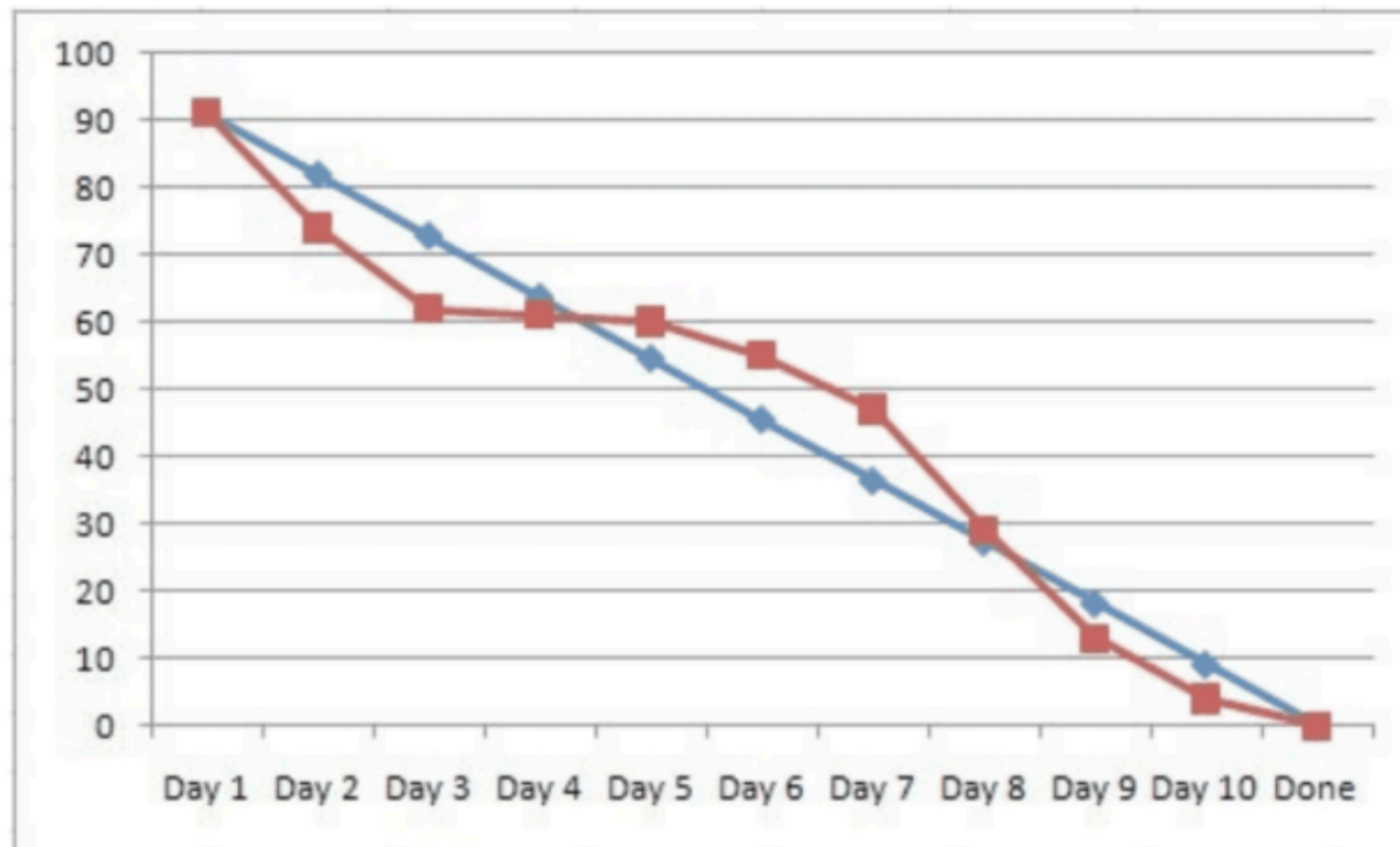
Source: Dave Snowden [www.cognitive-edge.com](http://www.cognitive-edge.com)

Use Cynefin to:

- Help understand how you perceive situations
- Make sense of other's behavior
- Help reach decisions
- Create a sense of place in a complex world

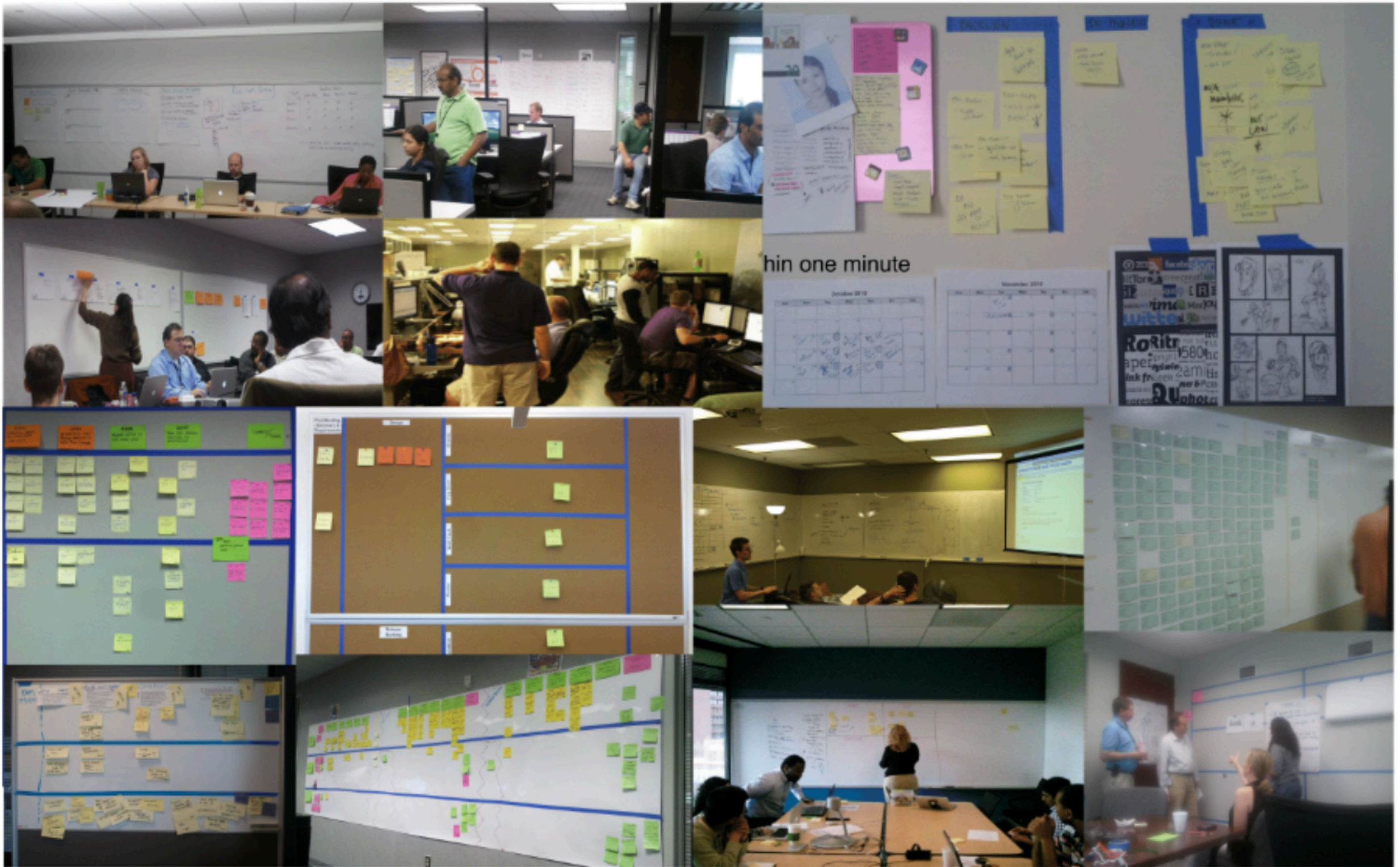
# Burn Down Charts

- **Days** on horizontal axis / **User story** units on vertical
- Does not track effort, only completed work [units]
- Theoretical trend line









# Collaborative Common Space + Info Radiators



<http://sloanreview.mit.edu/article/why-showing-your-face-at-work-matters/>



## Section Retrospective

<p>What Did I Like?</p> 	<p>What Surprised Me?</p> 
<p>What Ideas Were Sparked?</p> 	<p>My Takeaways</p> 

# Book Resources

- The Agile Pocket Guide, Peter Saddington, Wiley, 2012
- Agile Software Development with SCRUM by Ken Schwaber, Mike Beedle
- Agile Project Management with Scrum by Ken Schwaber
- User Stories Applied by Mike Cohn
- Agile Estimating and Planning by Mike Cohn
- Collaboration Explained, Jean Tabaka, Addison Wesley, 2006
- Agile Retrospectives, Esther Derby and Diana Larsen, Pragmatic Programmer, 2006
- Lean Software Development, Poppendieck and Poppendieck, Addison Wesley, 2003
- Agile & Iterative Development, Craig Larman, Pearson Education, 2004
- Agile Estimating and Planning, Mike Cohn, Pearson Education, 2005
- User Stories Applied, Mike Cohn, Pearson Education, 2004
- Extreme Programming Explained, Kent Beck, Addison Wesley, 2000
- Project Retrospectives, Norman Kerth, Dorset House, 2001
- The Knowledge Creating Company, Nonaka and Takeuchi, Oxford University Press, 1995
- The Pragmatic Programmer, Hunt and Thomas, Addison Wesley, 2001
- The Blind Men and the Elephant: Mastering Project Work, David Schmaltz, Berrett-Koehler Publishers; , March 2003
- Complexity and Management, Ralph D. Stacey, Routledge, 2000
- Corps Business, David Freedman, HarperCollins, 2000
- Industrial Dynamics, Jay W. Forrester, MIT Press, 1961
- The Art of Focused Conversation, Brian Stanfield, New Society Publishers, 2000