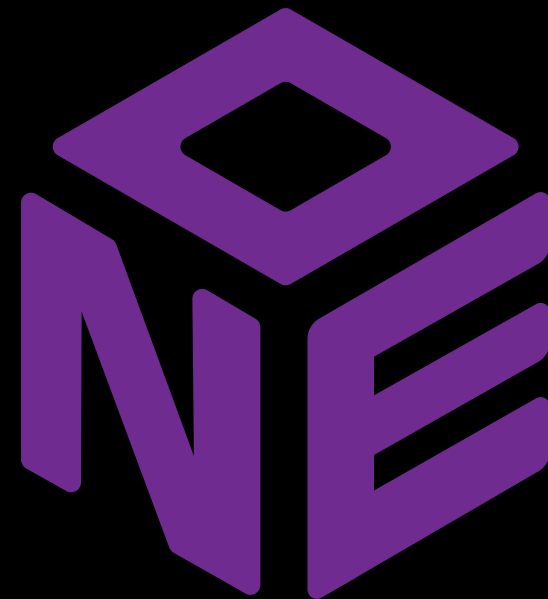




PMI ACP Preparation

Shiva Muthu



DXC

Agenda



1. Introduction	03
2. PMI ACP Introduction	04
3. Agile Introduction	08
4. Agile Mindset	23
5. Agile Principles	36
6. User Stories Applied	48
7. \$\$ Value Prioritization	68
8. Agile Planning and Estimation	84
9. Agile Methods and Metrics	95
10. Build High Performance Teams	133
11. Agile Retrospectives	151
12. PMI Code of Ethics	163

Shivashunmugam Muthu



- Live in Chennai, India
- Joined the company in 2006
- Two decades of Industry experience
- Account Delivery Leader in GIDC AMS Insurance Region
- 14 years experience in Project Management
- 8 years experience in Agile Project Management
- In good standing with PMP since 2008 and PMI-ACP since 2018

PMI ACP Introduction

PMI guidelines

Why certified?

- **PMI is the globally recognized organization for project management for several decades, since 1969**
- **JV certification by PMI and Agile Alliance**
- **Most respected and coveted title in the industry**
- **PMI-ACP is the fastest growing certification in the recent years**
- **Covers all agile methodologies like Scrum, Kanban, XP, TDD etcetera**
- **Confidence in undertaking the profession**
- **Career growth**
- **Increased Salability quotient both inside and outside**

Eligibility, Prerequisites and Price

Who Should Apply?

If you work on agile teams or if your organization is adopting agile practices, the PMI-ACP is a good choice for you. Compared with other agile certifications based solely on training and exams, the PMI-ACP is evidence of your real-world, hands-on experience and skill.

Gain and Maintain Your PMI-ACP

- › The certification exam has 120 multiple-choice questions and you have three hours to complete it.
- › To maintain your PMI-ACP, you must earn 30 professional development units (PDUs) in agile topics every three years.

Price

Member: US\$435.00

Non-member: US\$495.00

Prerequisites

- › 2,000 hours of general project experience working on teams. A current PMP® or PgMP® will satisfy this requirement but is not required to apply for the PMI-ACP.
- › 1,500 hours working on agile project teams or with agile methodologies. This requirement is in addition to the 2,000 hours of general project experience.
- › 21 contact hours of training in agile practices.

Reference Materials

- **No single bible like PMBOK**
- **Wide range of reference books published in PMI.org**
- **Practical experience is mandatory to answer the situational questions**
- **90% of the questions are situational to test your profession**

Agile Estimating and Planning

Author: Mike Cohn

Publisher: Pearson Education/Addison Wesley Professional

Agile Practice Guide

Publisher: Project Management Institute, Inc.

Agile Project Management: Creating Innovative Products

Author: Jim Highsmith

Publisher: Pearson Education/Addison Wesley Professional

Agile Retrospectives: Making Good Teams Great

Author: Esther Derby, Diana Larsen, Ken Schwaber

Publisher: Pragmatic Bookshelf

Agile Software Development: The Cooperative Game

Author: Alistair Cockburn

Publisher: Pearson Education

Coaching Agile Teams: A Companion for ScrumMasters, Agile Coaches, and Project Managers in Transition

Author: Lyssa Adkins

Publisher: Pearson Education/Addison Wesley Professional

Effective Project Management: Traditional, Agile, Extreme

Author: Robert K. Wysocki

Publisher: Wiley

Exploring Scrum: The Fundamentals

Author: Dan Rawsthorne with Doug Shimp

Publisher: CreateSpace Publishing

Kanban In Action

Author: Marcus Hammarberg, Joakim Sunden

Publisher: Manning Publications

Kanban: Successful Evolutionary Change for your Technology Business

Author: David J. Anderson

Publisher: Blue Hole Press

Lean-Agile Software Development

Author: Alan Shalloway, Guy Beaver, James R. Trott

Publisher: Pearson Education

User Stories Applied: For Agile Software Development

Author: Mike Cohn

Publisher: Pearson Education

Define “Agile”

Define agile



All Images Books News Shopping More Settings Tools



About 12,20,00,000 results (0.65 seconds)

Dictionary

Search for a word



agile

/ˈɑːdʒaɪl/

adjective

adjective: **agile**

- 1. able to move quickly and easily.
"Ruth was as agile as a monkey"

Similar: nimble lithe spry supple limber sprightly acrobatic

- able to think and understand quickly.
"his vague manner concealed an agile mind"

- 2. relating to or denoting a method of project management, used especially for software development, that is characterized by the division of tasks into short phases of work and frequent reassessment and adaptation of plans.
"agile methods replace high-level design with frequent redesign"

Origin



late Middle English: via French from Latin *agilis*, from *agere* 'do'.

Facts of “Agile”

Myths of “Agile”

- Agile is New
- Agile means no documentation
- Agile means no design
- Agile means no planning
- There is a right size for the user story
- Work must fit in a sprint
- Developers get to do what they like
- Scrum and Kanban are sworn enemies
- Agile doesn't work for fixed deadline projects
- Agile doesn't work for fixed brownfield projects

Myths

Why “Agile?”

Delivering Wrong – Video

<https://www.youtube.com/watch?v=WDoeEO6kSVEk>



How the customer explained it



How the project leader understood it



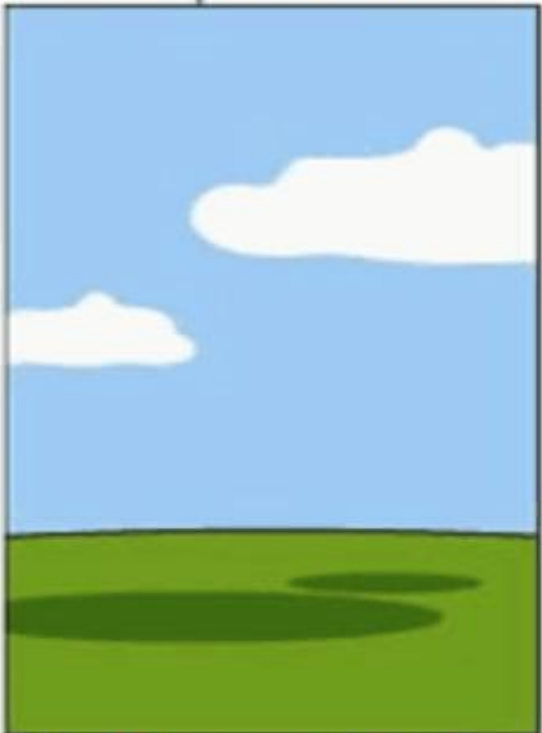
How the engineer designed it



How the programmer wrote it



How the sales executive described it



How the project was documented



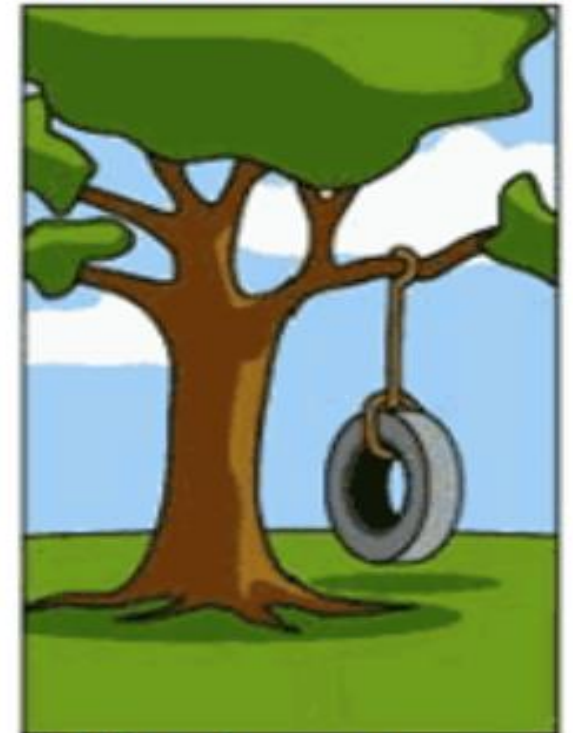
What operations installed



How the customer was billed

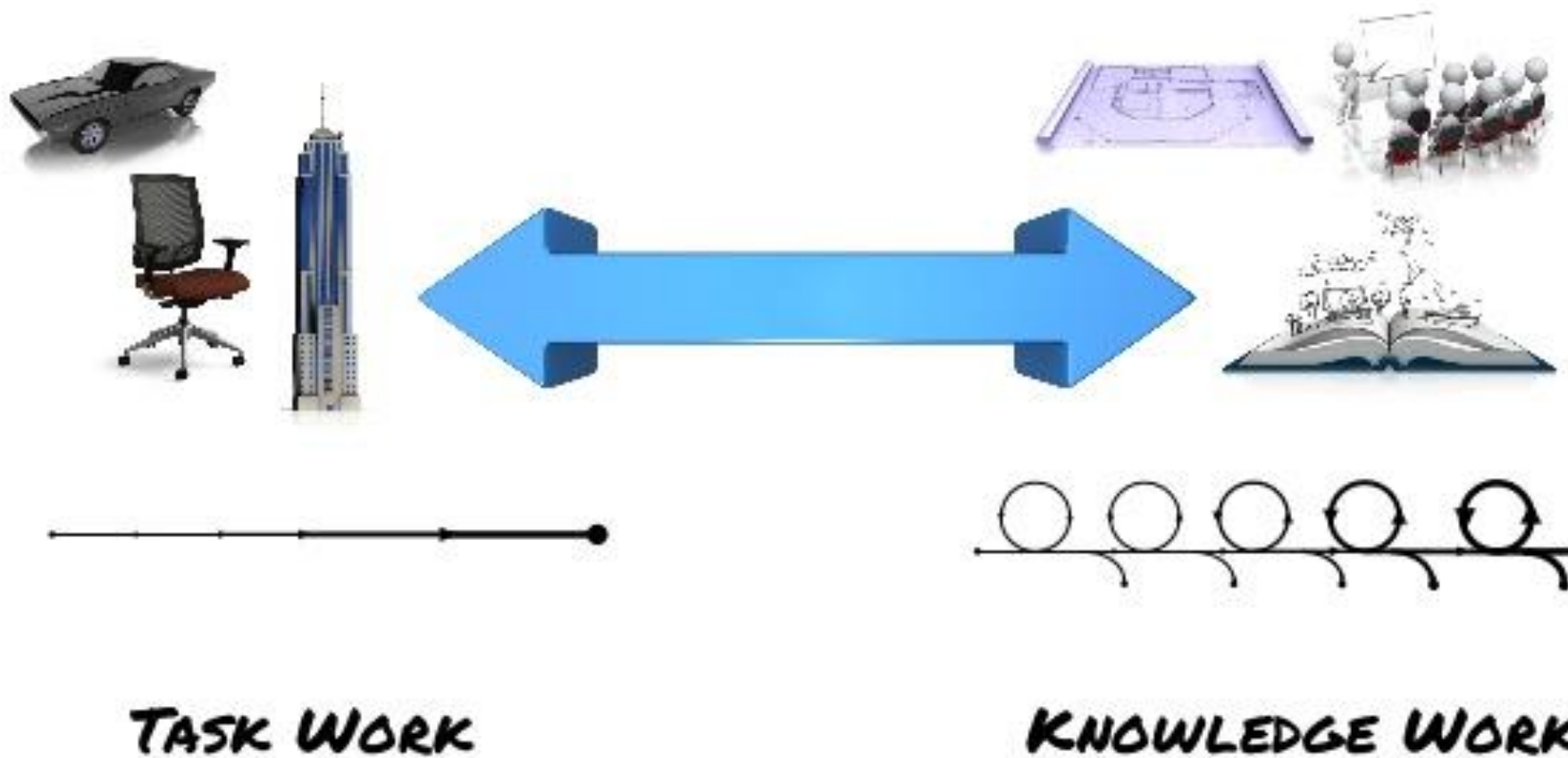


How the helpdesk supported it



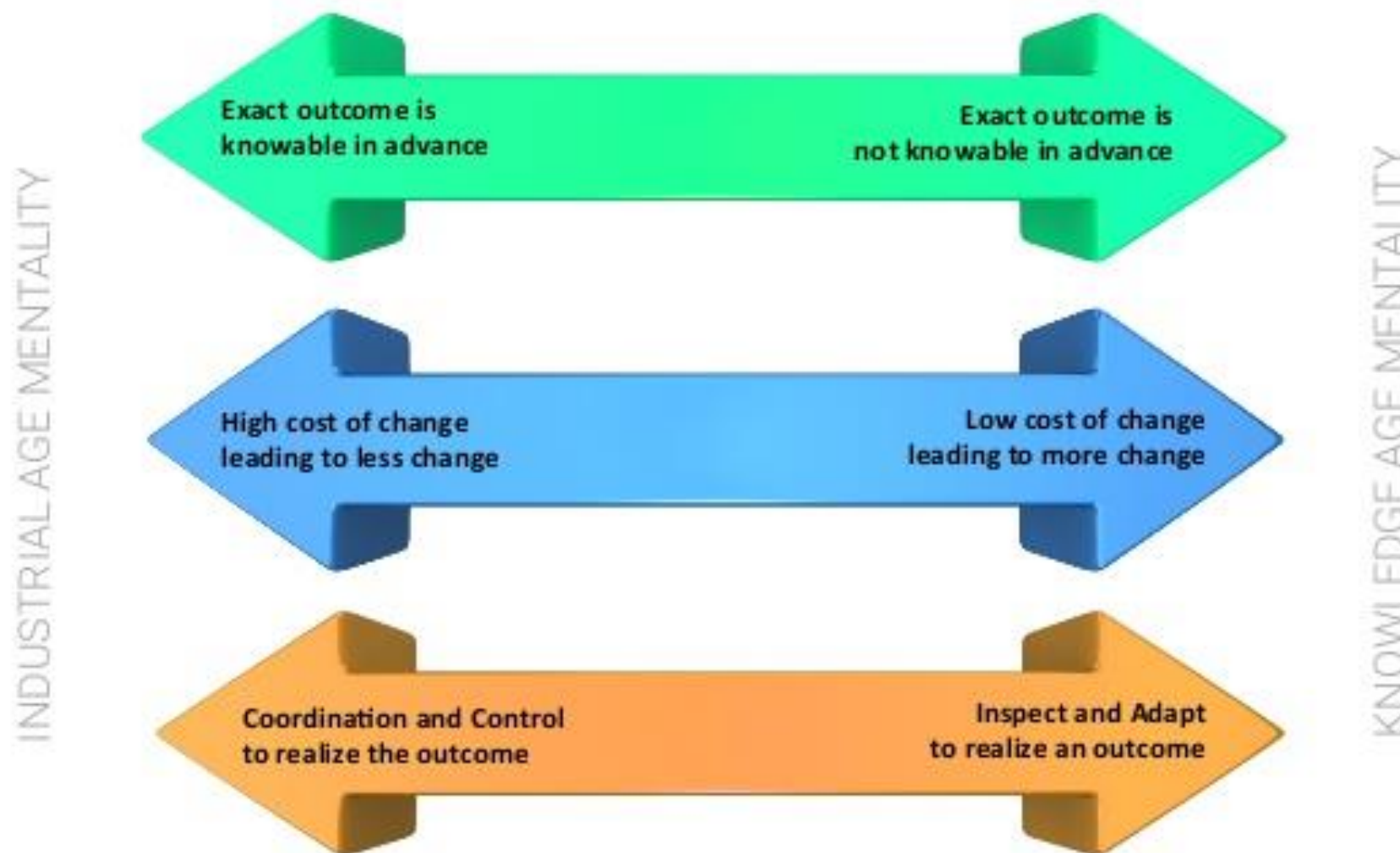
What the customer really needed

NOT ALL WORK IS THE SAME ...



**TASK
WORK**
(assembly line)

**KNOWLEDGE
WORK**
(creative work)





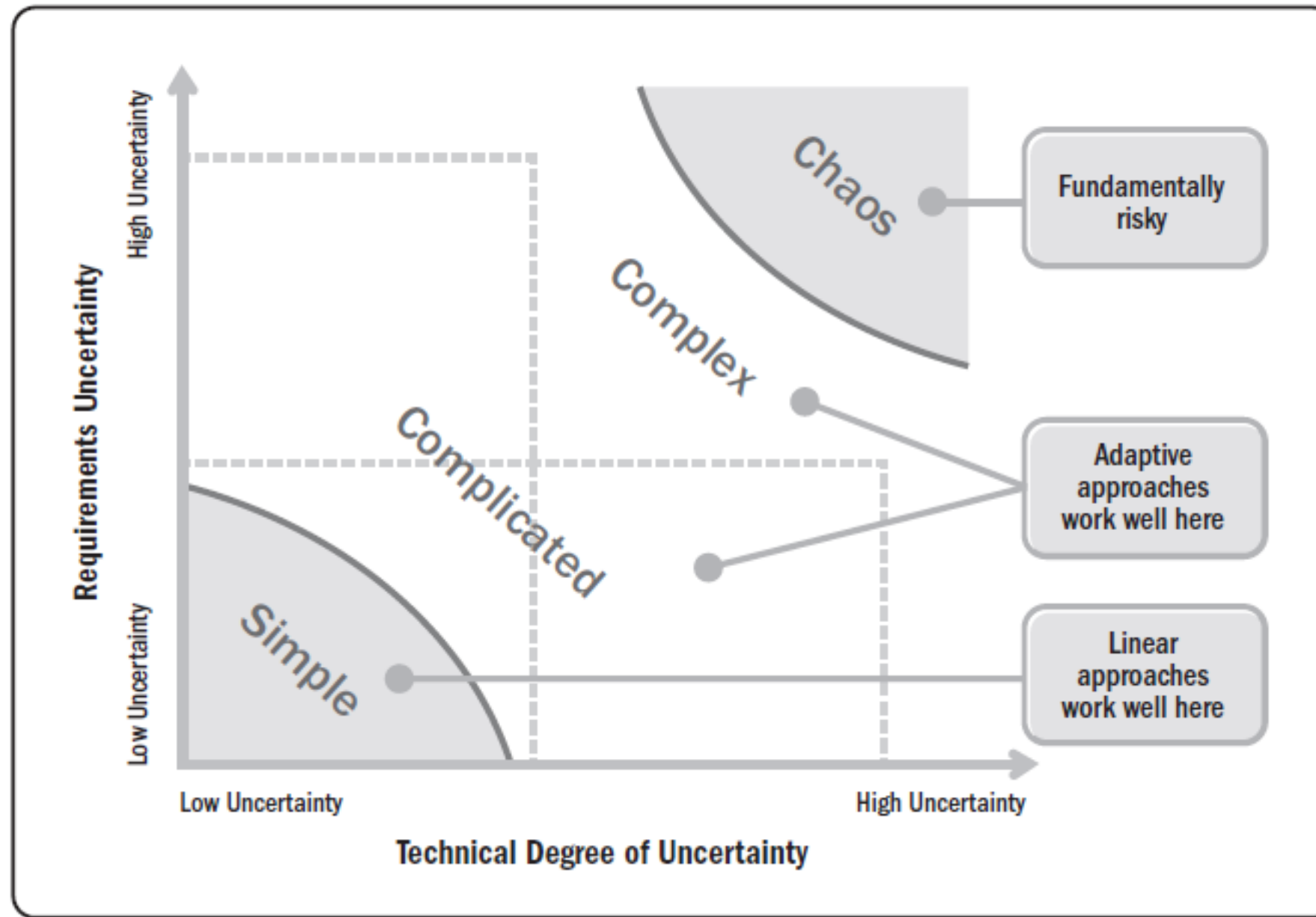




2-minute home

<https://www.youtube.com/watch?v=I2HqW-ABb20>

Stacey complexity model



How “Agile?” – Agile Mindset

Is the outcome “knowable” in advance?



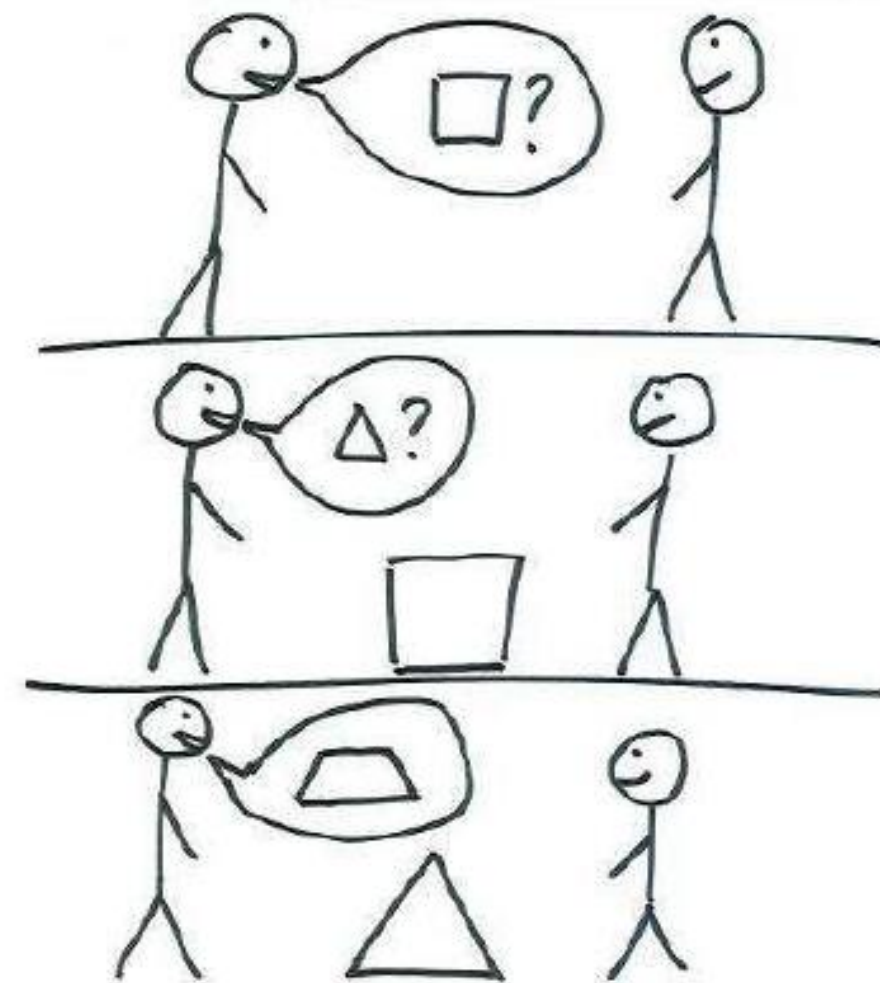
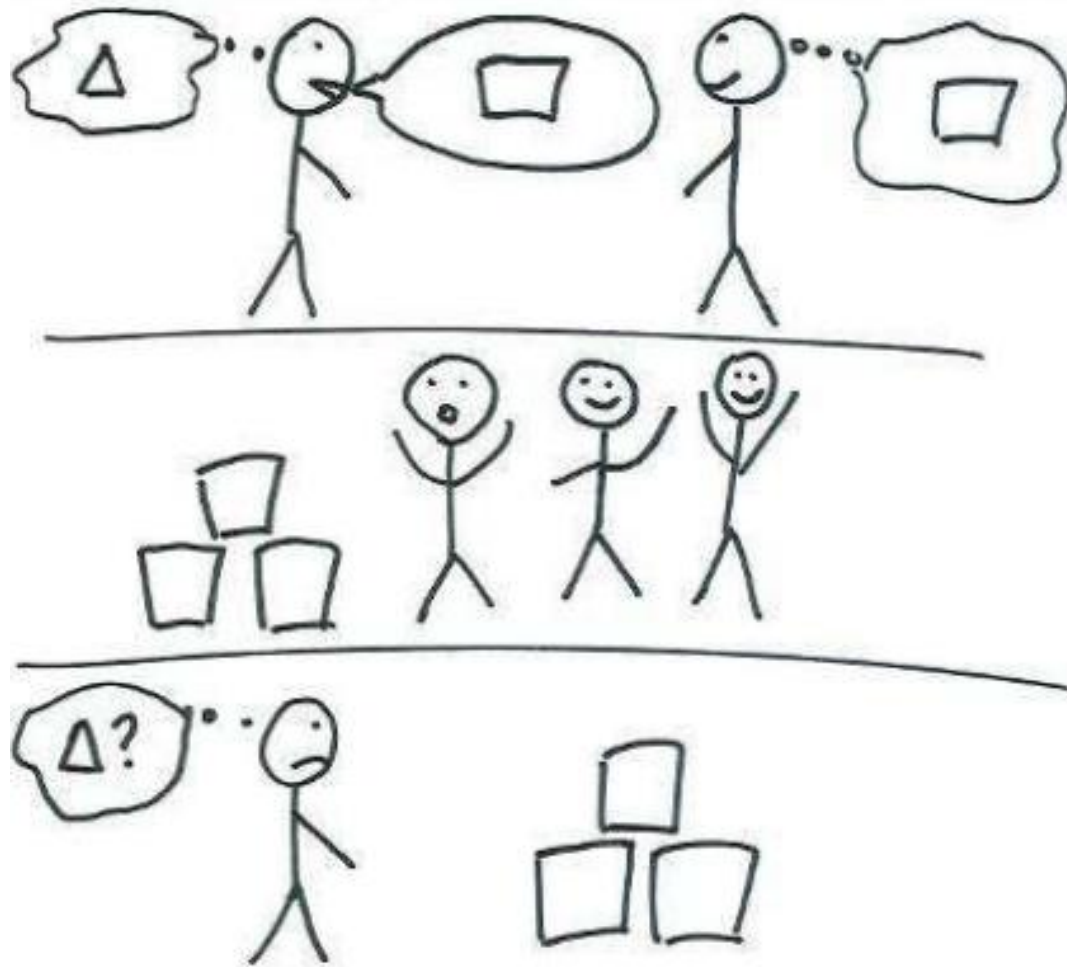
 <p>How the customer explained it</p>	 <p>How the customer understood it</p>	 <p>How the customer perceived it</p>	 <p>How the customer perceived it</p>	 <p>How the Business Consultant described it</p>
 <p>How the project was documented</p>	 <p>What operations included</p>	 <p>How the customer was billed</p>	 <p>How it was supported</p>	 <p>What the customer really needed</p>

IKIWISI

I'll Know It When I ~~S~~ee It

EXPERIENCE

IKIWISI: I'll Know It When I See It



Shu Ha Ri

https://www.youtube.com/watch?v=hF_VC-wvSgQ

Shu-Ha-Ri

Applied to Agile team

守破離

Stage	Iteration Planning	Release Planning	Daily stand-up	Velocity	Retrospectives	Release frequency
SHU	Team struggles with the process, has trouble defining task & duration	Team is not sure what it will be doing next iteration	Lots of off-topic discussion, resembles a status meeting	Velocity is unpredictable, it's up, it's down from sprint to sprint	Team seems to be going through the motion on the Retro	Team struggles to get working software out the door every sprint
HA	Team is able to do iteration planning in a time box	Team knows what it will be working on 2-3 iterations into the future	Everyone is participating and the 3 basic questions are being addressed	Velocity growth trend is increasing for three sprints in a row	Team has positive discussions aligned with Agile Manifesto themes and values	Most sprints result in a good build with occasional build issues
RI	Team is identifying tasks and durations in advance and meeting is fast and efficient	Team knows what it will be working on 3 or more iterations out into the future	Executed with precision, nothing extraneous, transparency & truth	Velocity growth trends slows, levels off, is consistent & predictable	Team is instituting meaningful process improvement every sprint	Every sprint results in a good build of working software, no exceptions

Using Shu-Ha-Ri in Agile

SHU	HA	RI
1. Customer focused	1. Satisfy focused	1. Delight customer
2. Welcome changes	2. Embrace changes	2. Seek changes
3. Deliver regularly	3. Deliver frequently	3. Deliver continuously
4. Engage business people	4. Being a hole team	4. Live as a hole team
5. Hire the right people	5. Motivate people	5. Trust people
6. Talk face-to-face	6. Talk mind-to-mind	6. Talk heart-to-heart
7. Measure output	7. Measure working software	7. Measure value delivered
8. Maintained pace	8. Maintained pace indefinitely	8. Maintained pace and rhythm
9. Quality focused	9. Excel at quality	9. Excel at quality & get things done
10. Keep it simple	10. Less is more	10. Simplicity is the ultimate sophistication
11. Self-organizing team	11. Delegation board	11. Remove management
12. Team retrospective	12. Personal retrospective	12. Company retrospective

Agile Fundamentals: Shu-ha-ri applied to Agile team

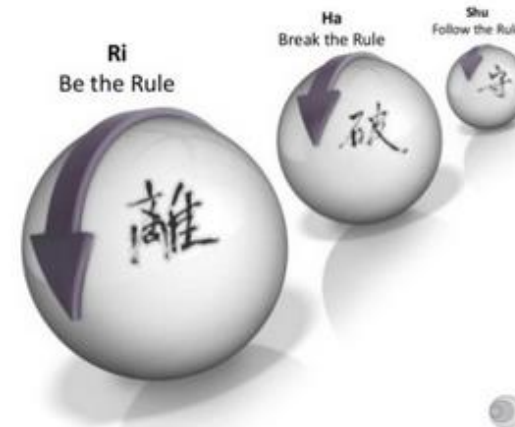


Source: <http://agilean.dk/agile-maturity-model-shu-ha-ri/>
Agile Fundamentals: Shu-ha-ri applied to Agile team



Shu – Imitate
Ha – Integrate
Ri - Innovate

1. Shu – 2. Ha – 3. Ri





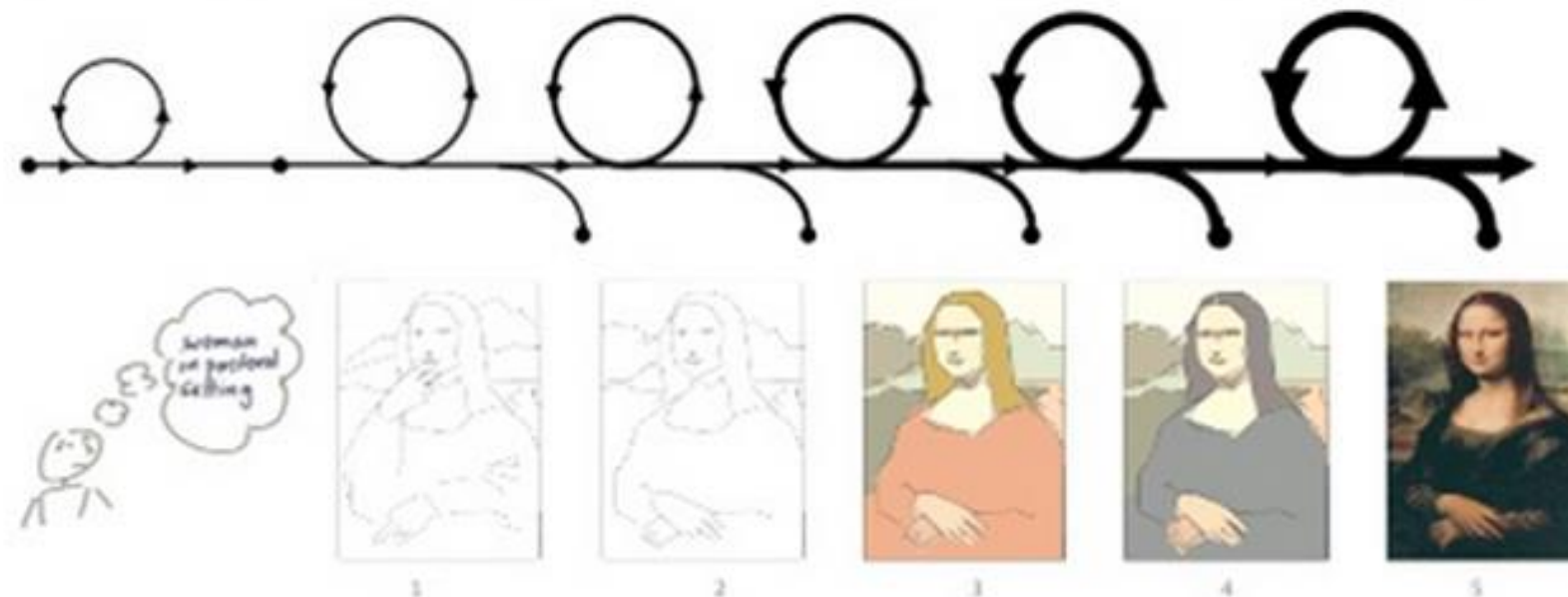
Fixed Mindset approach to delivery (Assembly Line)

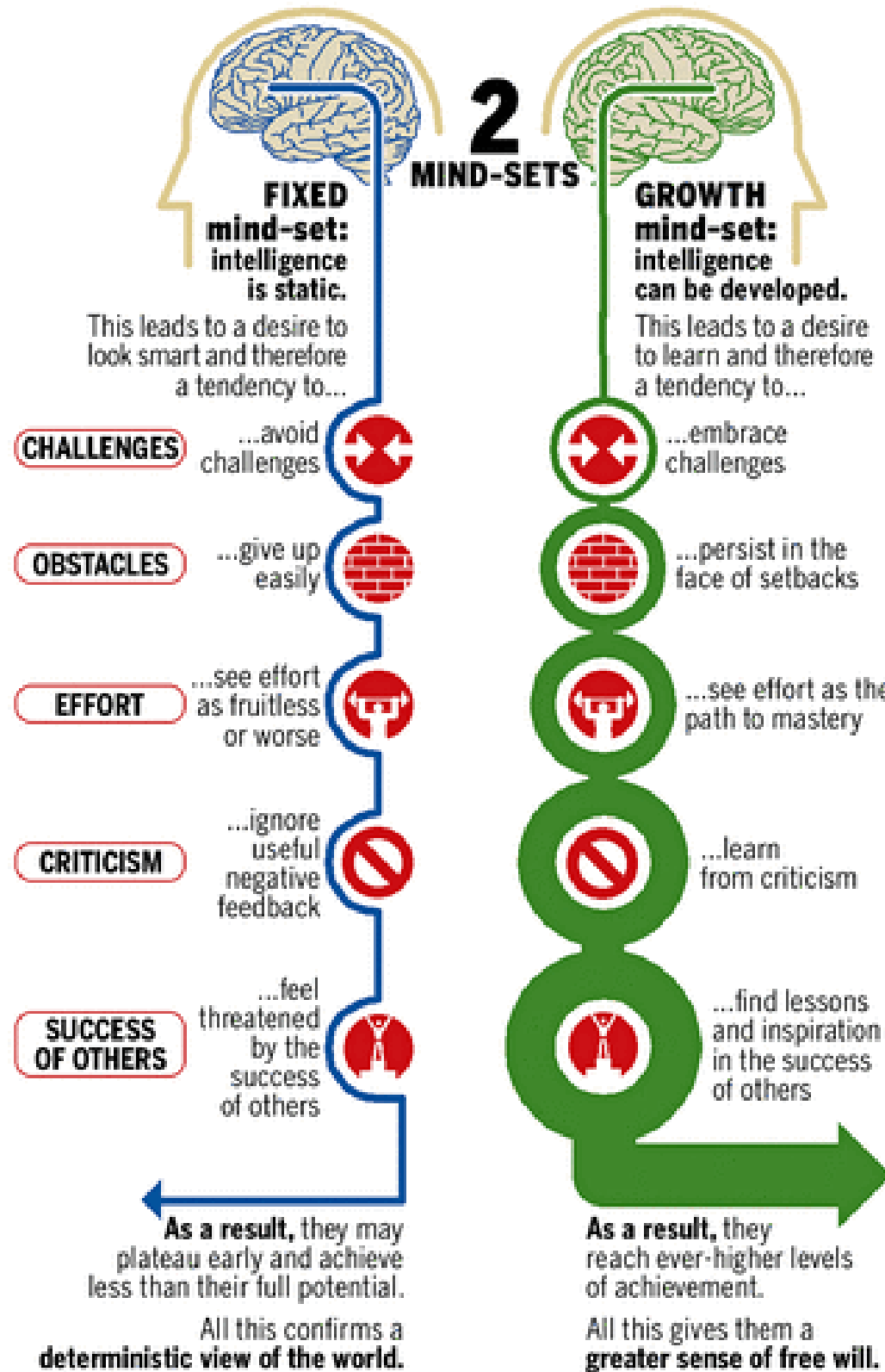
Must "nail down" the output in order to start delivery (Linear Thinking)



Agile Mindset approach to delivery (Knowledge Work)

Discover and learn through valuable output and welcoming change (Circular Thinking – IKIWISI)





https://www.youtube.com/watch?v=KUWn_TJTrnU

10 Growth Mindset Statements

FIXED MINDSET



What can I say to myself?

INSTEAD OF:

I'm not good at this.

I'm awesome at this.

I give up.

This is too hard.

I can't make this any better.

I just can't do Math.

I made a mistake.

She's so smart. I will never be that smart.

It's good enough.

Plan "A" didn't work.

TRY THINKING:

1 What am I missing?

2 I'm on the right track.

3 I'll use some of the strategies we've learned.

4 This may take some time and effort.

5 I can always improve so I'll keep trying.

6 I'm going to train my brain in Math.

7 Mistakes help me to learn better.

8 I'm going to figure out how she does it.

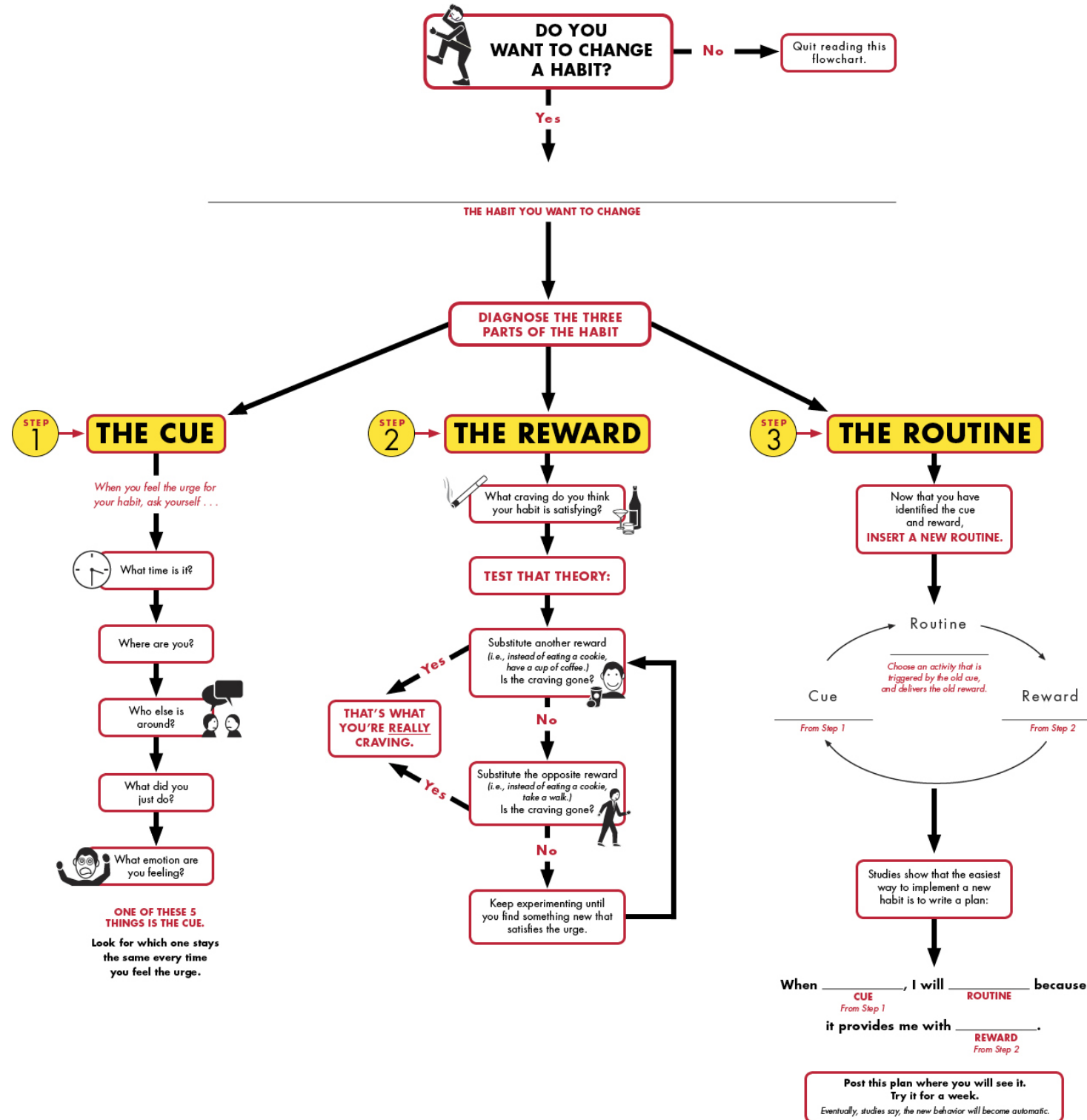
9 Is it really my best work?

10 Good thing the alphabet has 25 more letters!

GROWTH MINDSET



HOW TO CHANGE A HABIT



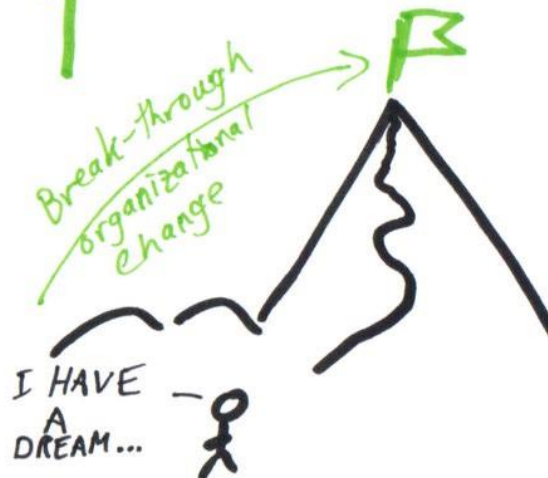
DOING AGILE PRACTICES \neq BEING AGILE MINDSET



"Scrum, BUT..."
"Cargo cult Agile"



PROBLEM-DRIVEN
ADOPTION OF
PRACTICES



ADOPTION

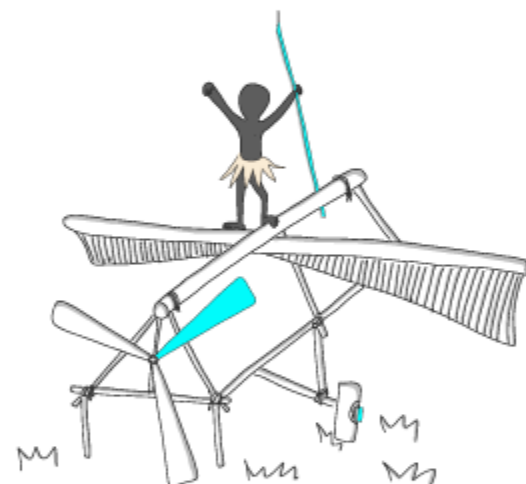
TRANSFORMATION

CC 2012 MICHAEL SANOTA

CARGO CULT By Matt & Nat

AGILE KNOWLEDGE BASE SKETCH #4

DOING AGILE



BEING AGILE



A COGNITIVE BIASES

DEVIATIONS FROM NORM OR RATIONALITY IN JUDGMENT. INDIVIDUALS CREATE THEIR OWN „SUBJECTIVE SOCIAL REALITY“ FROM THEIR PERCEPTION OF THE INPUT.

LOGICAL FALLACIES

BAD REASONING CAUSED BY WRONG ASSUMPTIONS OR MISCONCEPTIONS.

© SketchingPM.com



STOP DOING AGILE AND START BEING AGILE.

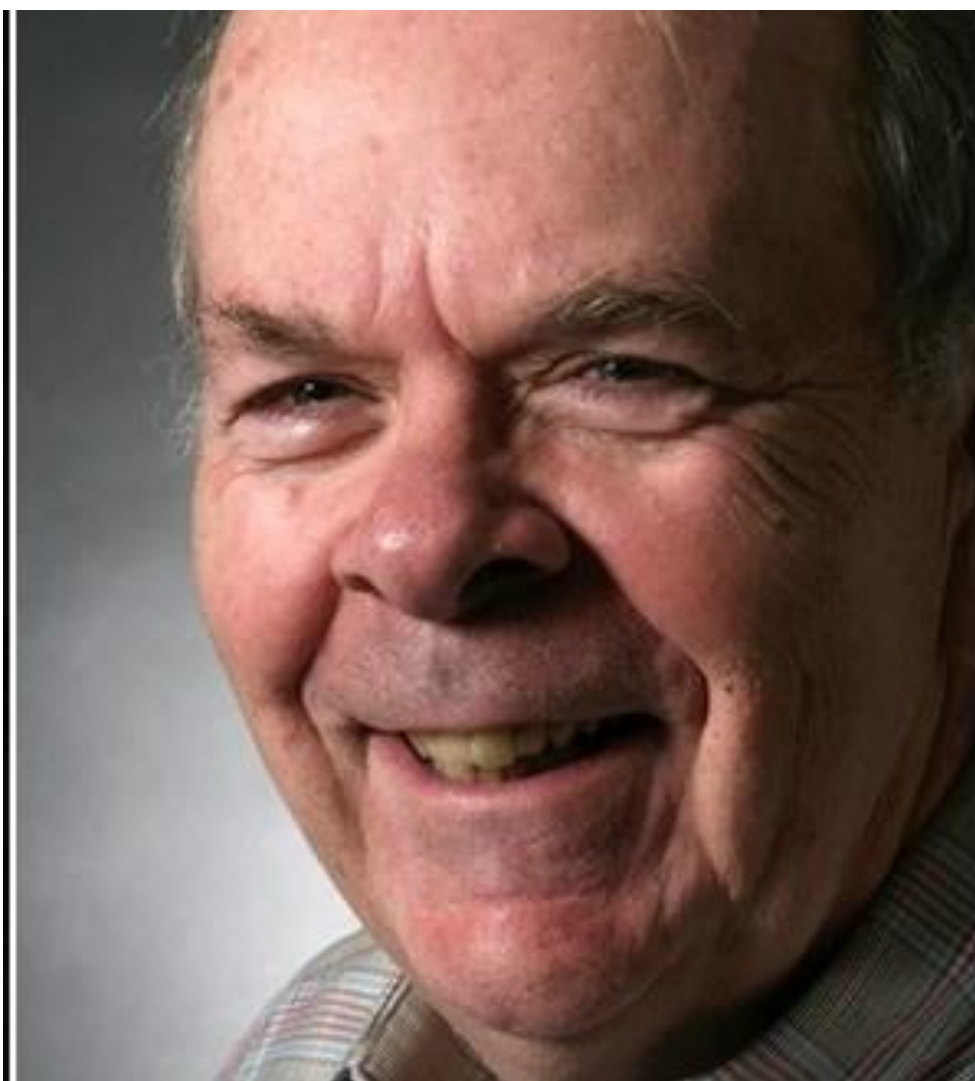
UNKNOWN



Agile is an attitude, not a technique with boundaries. An attitude has no boundaries, so we wouldn't ask 'can I use agile here', but rather 'how would I act in the agile way here?' or 'how agile can we be, here?'

— *Alistair Cockburn* —

AZ QUOTES

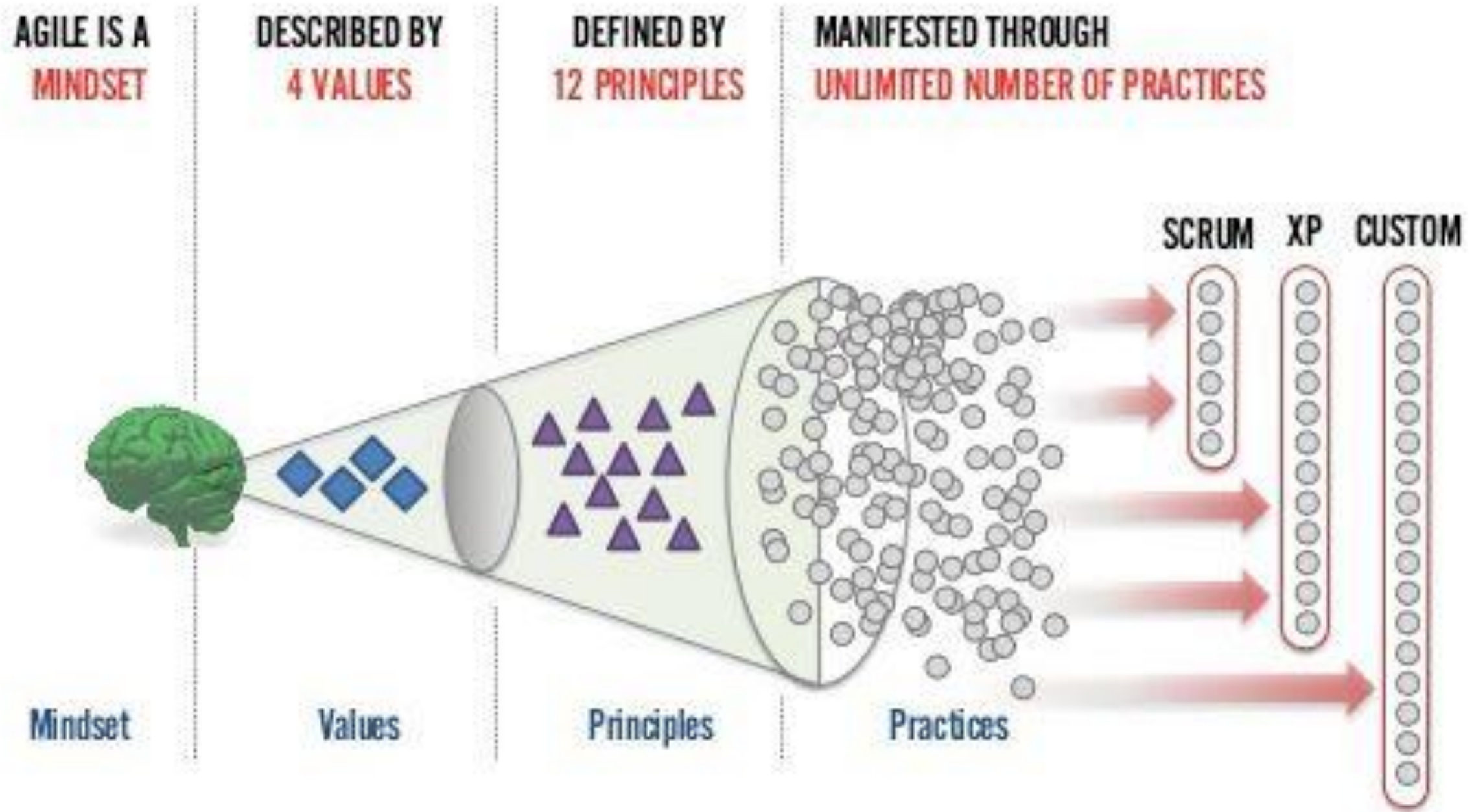


Agility is the ability to adapt and respond to change ... agile organizations view change as an opportunity, not a threat.

— *Jim Highsmith* —

AZ QUOTES

Agile Principles



Manifesto for Agile Software Development



We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Manifesto by Agile Alliance



We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

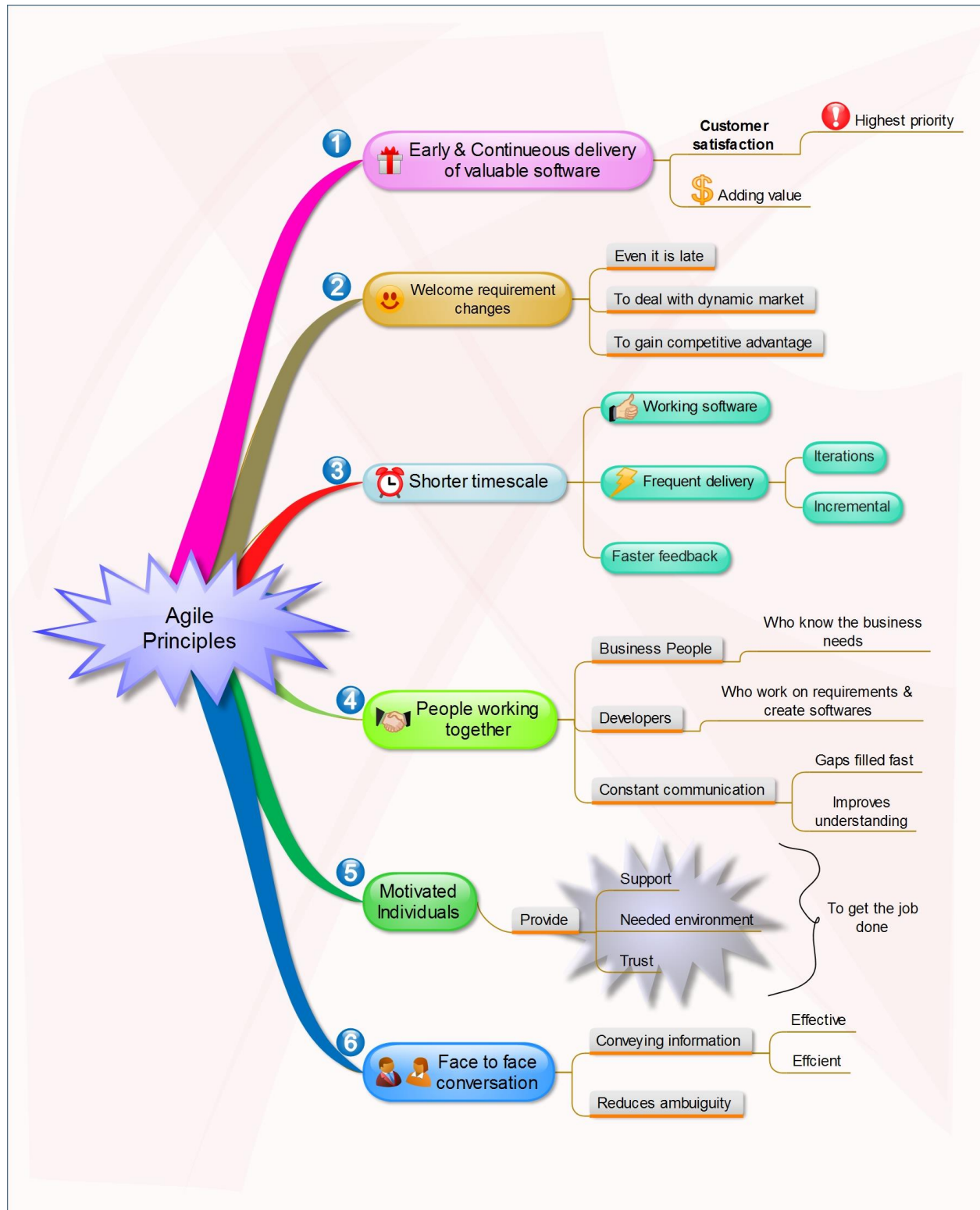
Responding to change over following a plan

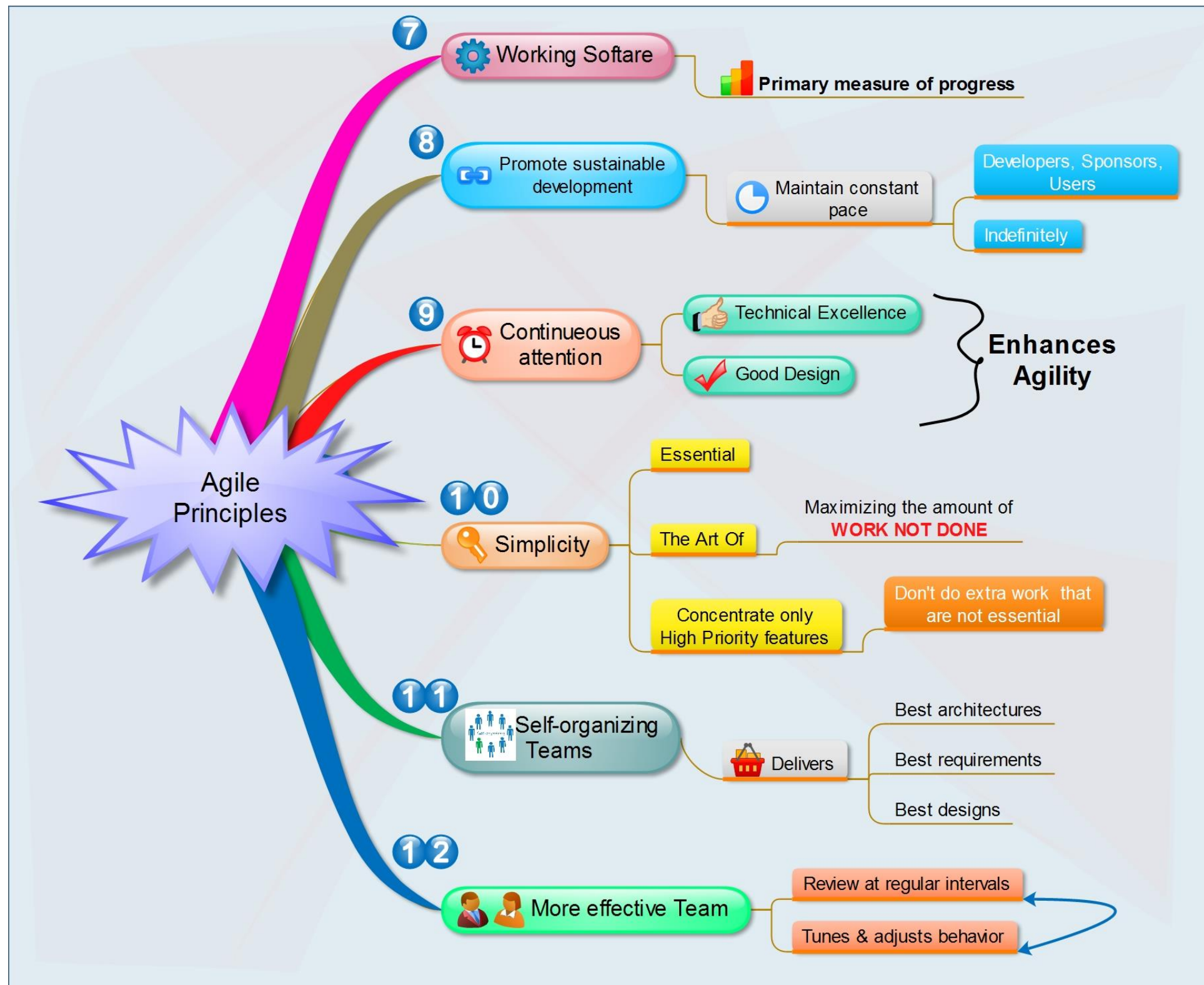
That is, while there is value in the items on the right, we value the items on the left more.

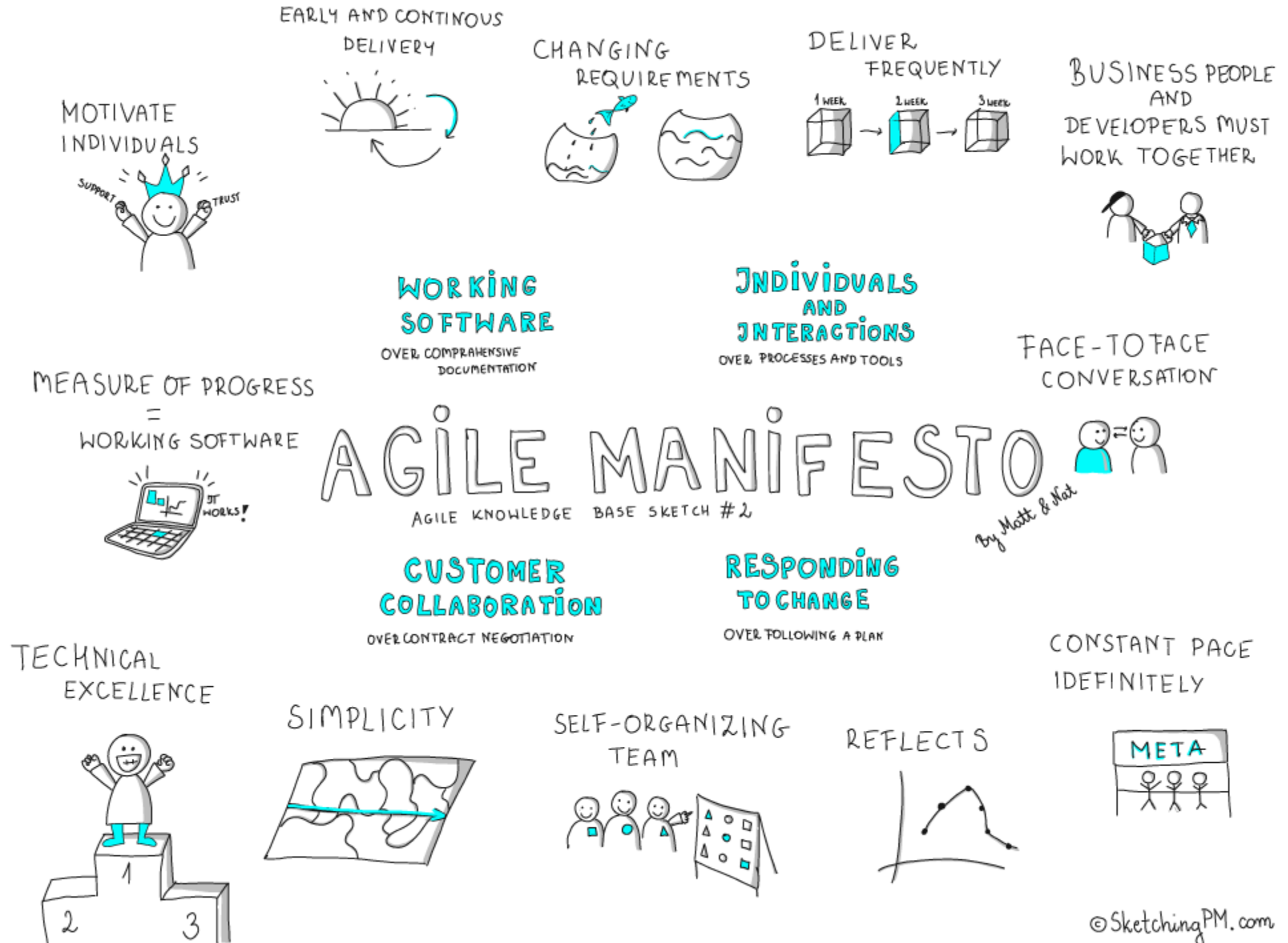
12 Principles



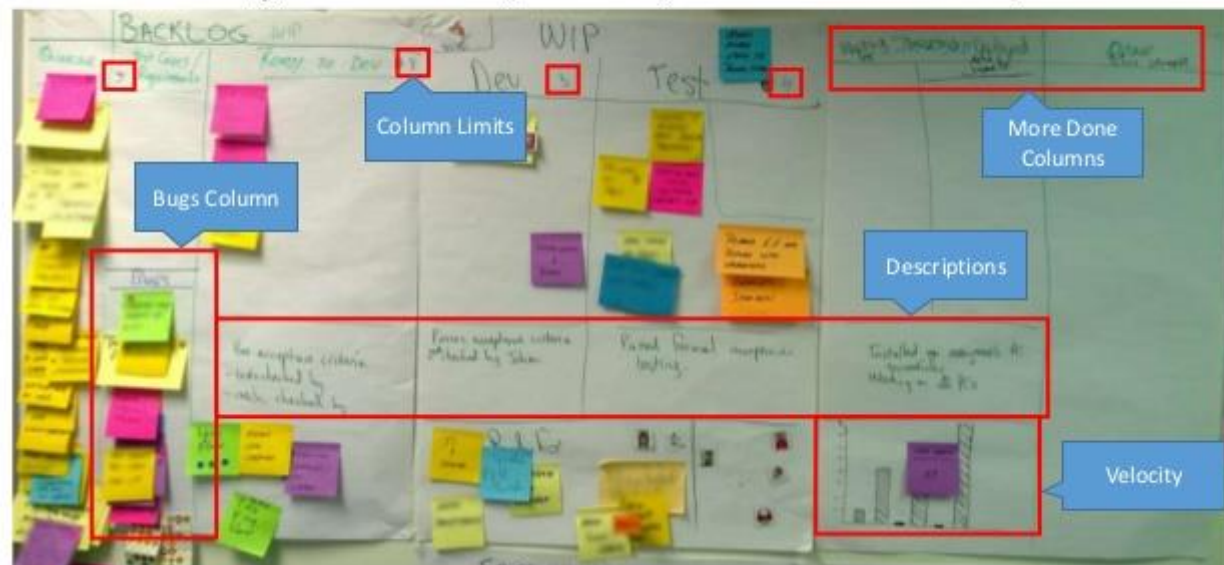
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of 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.







Agile Board (Scrum, Kanban ... etc)



BEWARE
CROCODILES
IN
MEETINGS.
(Big mouth,
small ears)



TASKS



Domain I	Agile Principles and Mindset
Task 1	Advocate for agile principles by modeling those principles and discussing agile values in order to develop a shared mindset across the team as well as between the customer and the team.
Task 2	Help ensure that everyone has a common understanding of the values and principles of agile and a common knowledge around the agile practices and terminology being used in order to work effectively.
Task 3	Support change at the system or organization level by educating the organization and influencing processes, behaviors, and people in order to make the organization more effective and efficient.
Task 4	Practice visualization by maintaining highly visible information radiators showing real progress and real team performance in order to enhance transparency and trust.
Task 5	Contribute to a safe and trustful team environment by allowing everyone to experiment and make mistakes so that each can learn and continuously improve the way he or she works.
Task 6	Enhance creativity by experimenting with new techniques and process ideas in order to discover more efficient and effective ways of working.
Task 7	Encourage team members to share knowledge by collaborating and working together in order to lower risks around knowledge silos and reduce bottlenecks.
Task 8	Encourage emergent leadership within the team by establishing a safe and respectful environment in which new approaches can be tried in order to make improvements and foster self-organization and empowerment.
Task 9	Practice servant leadership by supporting and encouraging others in their endeavors so that they can perform at their highest level and continue to improve.



- Information Radiators
- Safe Environment
- Servant Leadership
- Self Organizing
- Simplicity
- Celebrating Success
- Enhances Agility
- Business and Developers work together
- Maintain pace indefinitely
- Welcome change
- Continuous improvement
- F2F interaction is the desired way of communication
- Continuous attention to Technical excellence
- Customer satisfaction
- Frequent delivery – Iterative/Incremental
- Working software is a measure of progress
- Lean methodology
- Don't be crocidiles



WE DON'T NEED AN ACCURATE DOCUMENT,
WE NEED A SHARED UNDERSTANDING

User Stories Applied

USER STORIES

AGILE KNOWLEDGE BASE SKETCH #7

By Matt & Nat

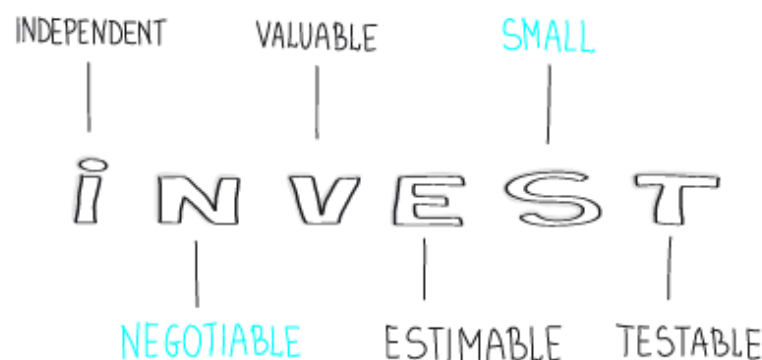
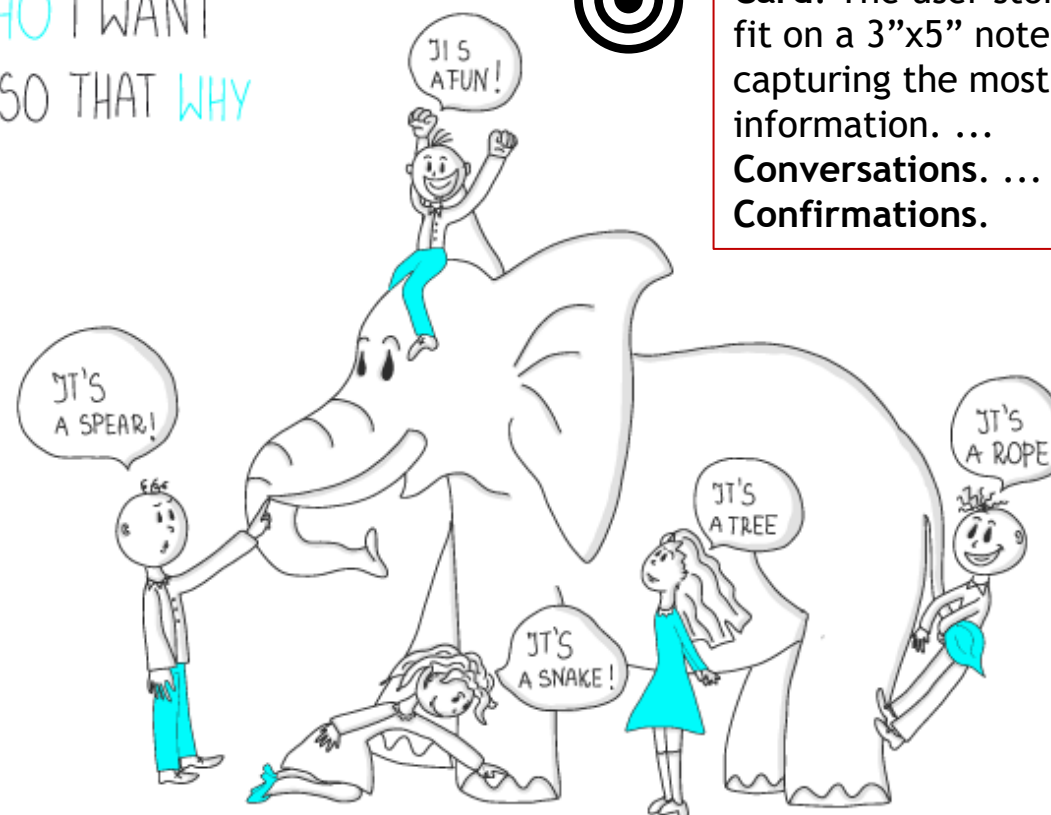
A good user story consists of three elements, commonly referred to as the **three C's**:

- Card.** The user story should be able to fit on a 3"x5" note card, efficiently capturing the most important information. ...
- Conversations.** ...
- Confirmations.**

AS WHO I WANT
WHAT SO THAT WHY



AS A USER, I CAN LOGIN TO THE SHOP
IN ORDER TO BUY SOME STUFF LATER



mnemonic

© SketchingPM.com

INVEST

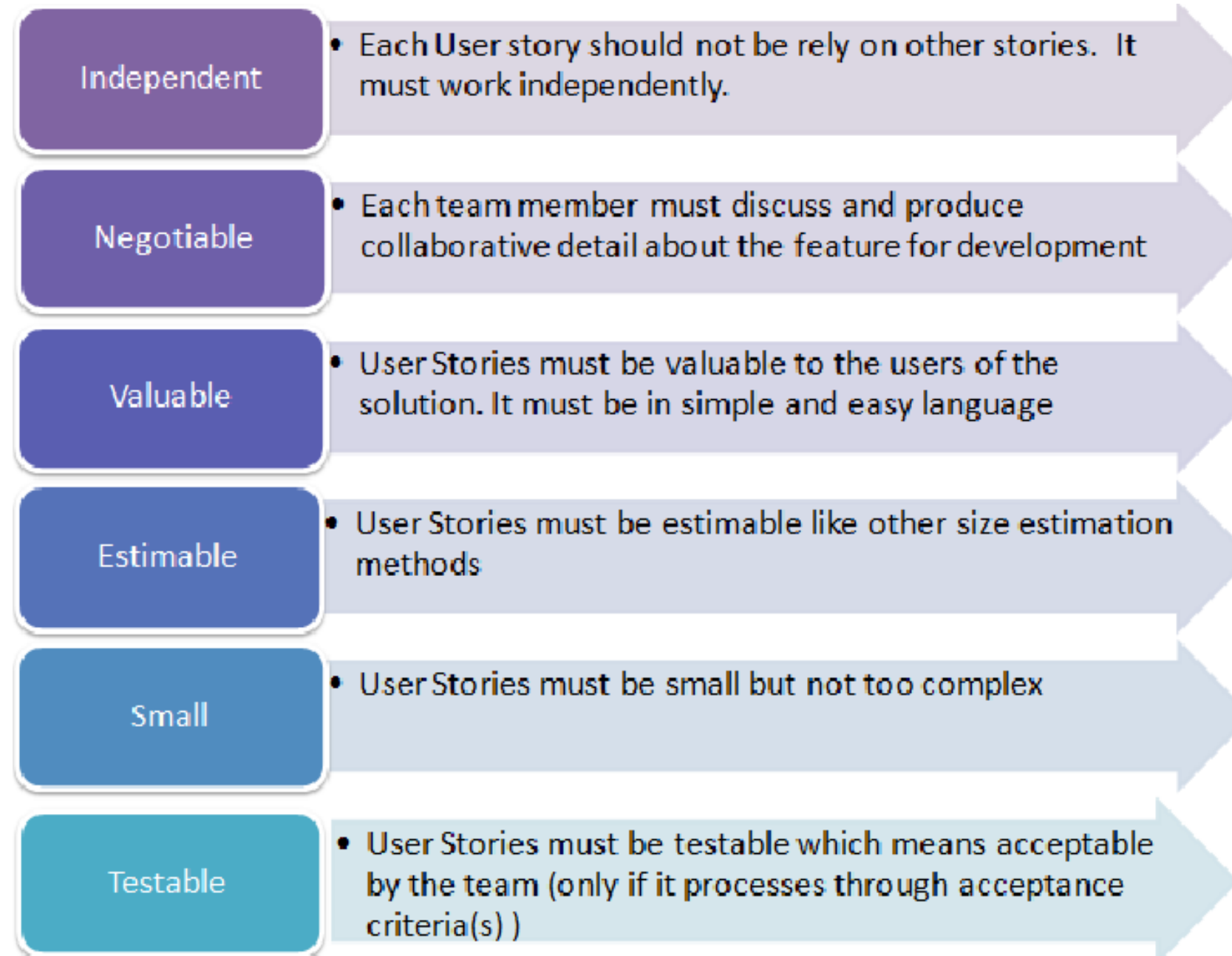


Figure 2.INVEST Principle [8]

Business Analysts and Development team work together to create

Examples



A bank customer can change his PIN.

Acceptance Criteria:

As a student, I can find my grades online so that I don't have to wait until the next day to know whether I passed.

Acceptance Criteria:

One level of undo

Acceptance Criteria:

As a book shopper, I can read reviews of a selected book to help me decide whether to buy it.

Acceptance Criteria:

As an author, I want the spell checker to ignore words with numbers so that only truly misspelled words are indicated.

Acceptance Criteria:

Counter Examples



“Design brochure layout.”

Drawbacks: **not Independent, no business Value**. This is a task representing a horizontal architectural layer or phase. The architecture will be done in a vacuum, possibly contributing to analysis paralysis.

Better: “As a dog owner, I can find a meal schedule on the brochure so I know whether this doggy day care center is appropriate for my hungry dog.”

This will lead to only the necessary amount of design to support this Sprint’s features. The layout might change the next Sprint, but rework is cheaper than no work.

“Write game rules.”

Drawbacks: **not Independent, no business Value, not Small**.

Better: “As a newbie game player, I want to know who goes first so we can start the game.”

Better: “As a competitive gamer, I want a way to leapfrog my opposing players.”

“I want the brochure to be colorful.”

Drawbacks: **not Independent, not Estimable** (without knowing other features of brochure), not Small.

This is an easy trap for those of us who grew up with the habit of writing “the JFIDM _shall_ comply with the IEEE-488 interface specification.”

Better: Use “colorful” and other cross-cutting requirements as acceptance criteria on each of the specific features in the backlog they apply to.

Contd..



“As Product Owner, I want a list of highly-rated restaurants on the brochure.”

Drawbacks: **It's not only about you!**

Better: Focus on your end users and stakeholders. “As a gourmet tourist, I want a list of highly-rated restaurants on the brochure.”

Better: “As the Chicago Public Health Department, I want warnings about restaurants that serve raw ingredients so that tourists don't get sick on our dime.”

Play test the game.”

Drawbacks: **Not Independent.** Encourages phase wise development.

Better: Make testing, refactoring, etc. a default acceptance criteria on every Product Backlog Item.

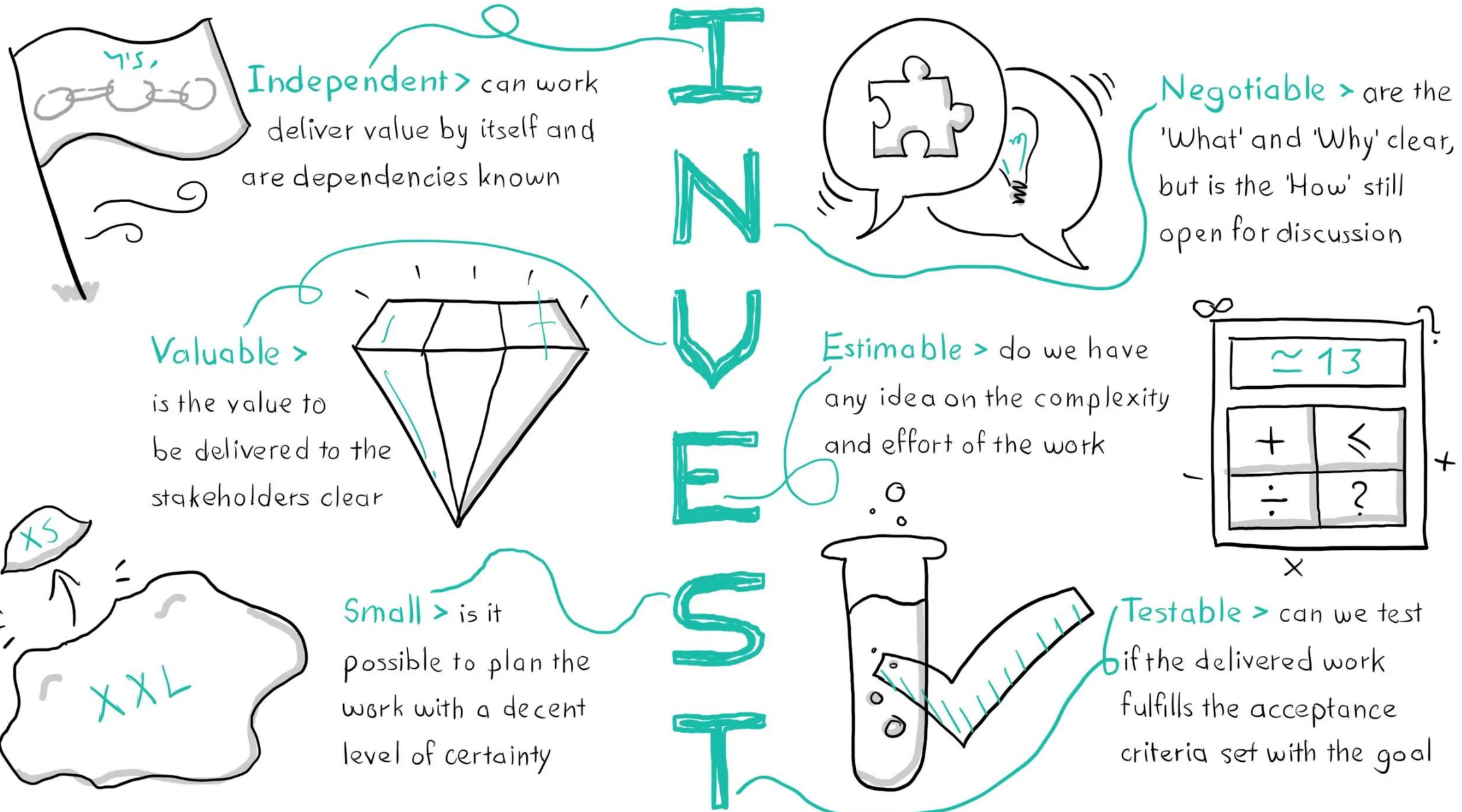
But: If you failed to fully test and refactor in previous Sprints, you are in technical debt! You are already working on a legacy product. In this case you may need to make testing and refactoring first-class Product Backlog Items to make up for your sins. This practice is controversial, and technical debt repayment cannot honestly be called a User Story. A PBI that's not a User Story may still be useful as a starting point for a conversation about how to reduce technical debt incrementally while continuing to deliver new value.

Definition of Ready

Idea by Bill Wake
Read more: agilety.com/DoR

Try to INVEST in not needing a Definition of Ready!

@jordanngross



What does it mean to be **READY**?

1. Defined clearly enough that all members of the team understand what must be done
 - Includes team-developed tasking, if needed
 - Assume some ongoing discussion to refine, coordinate and clarify
2. Includes clear statement of resulting business value that allows the Product Owner to prioritize
3. Includes any required enabling specs, wire frames, etc.
4. Fully meet INVEST criteria for user stories
 - Estimated and sized to complete easily within one sprint
5. Free from external dependencies
 - I.e. there is nothing beyond the team's control that must be done first in order to complete the story

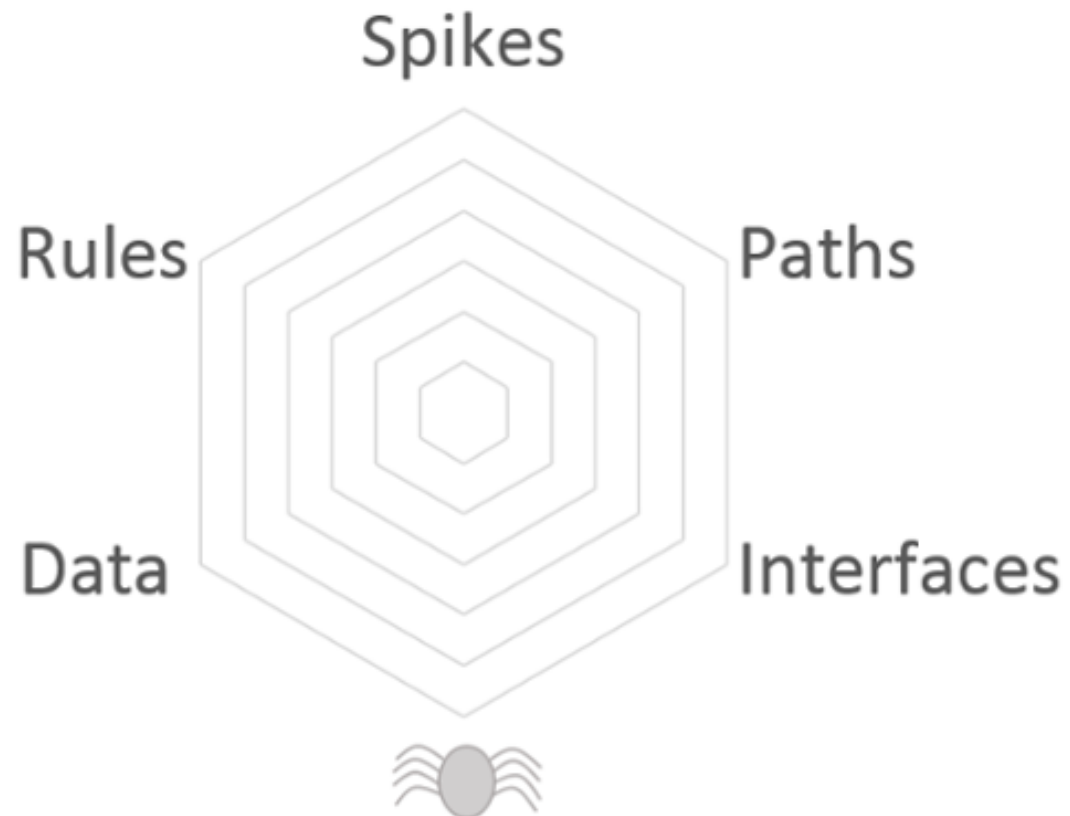
What Does it Mean to be DONE?

1. “Definition of Done” (DoD) decided on beforehand – along with acceptance tests
 - DoD can be standard across a group of common stories, or defined specifically for unique ones
2. Done means the feature has been developed, tested AND meets all required acceptance tests
3. Ideally, Done means the feature could be shipped to a customer
4. Product Owner officially “Accepts” Done features back from Team at the Sprint Review meeting

User Stories - Split

mnemonic

SPIDR



Story Splitting

Why Split User Stories?



the S.P.I.D.R. approach to splitting stories



SPIKES

Make a large story smaller by pulling out a spike, which is a research activity after which the team will know more.

- Sometimes just doing a spike makes the remaining work a manageable size.
- Other times, the new knowledge created by the spike makes it easier to see ways to split the story.

RULES

Sometimes a story is large because of the business rules, technology standards, or such that must be supported.

- Consider relaxing support for these rules in an initial story.
- Add support for additional rules in subsequent stories.

PATHS

Consider the paths through a story and split each path into its own story.

- Draw a simple flowchart of what happens in a story. Each sequence of steps can be a story.
- Expand one big step of the flowchart into a story.

DATA

Look for ways to split the story based on the type of data that must be supported.

- Can a first story support valid data and a later story add support for invalid data?
- How about frequent types of data and less frequently seen types of data?

INTERFACES

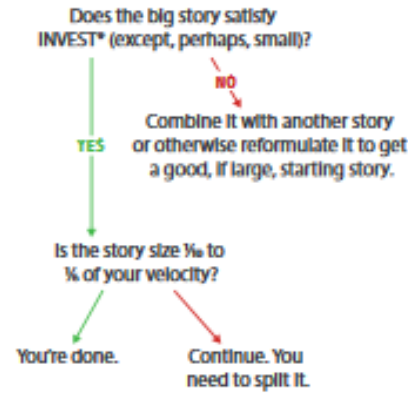
Split a story across multiple interfaces if supporting those interfaces makes the story take significantly longer to develop.

- Split out stories by browser type or version, or by different hardware.
- Consider building a minimal user interface first or leave styling out of an interface initially.

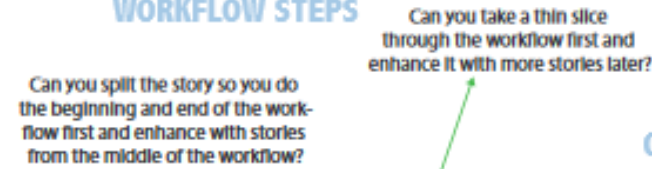


HOW TO SPLIT A USER STORY

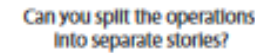
1 PREPARE THE INPUT STORY



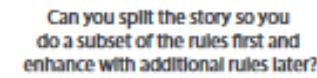
WORKFLOW STEPS



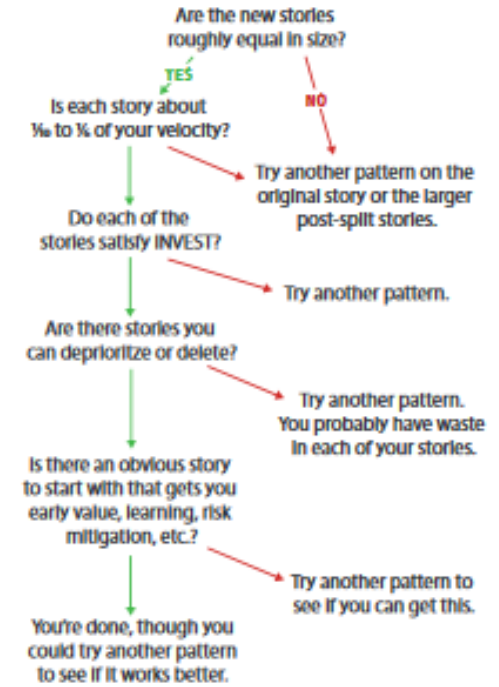
OPERATIONS



BUSINESS RULE VARIATIONS

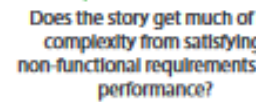
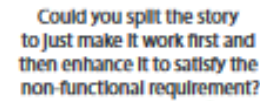


3 EVALUATE THE SPLIT

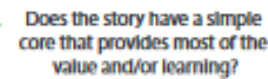
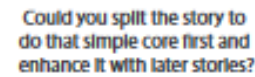


2 APPLY THE SPLITTING PATTERNS

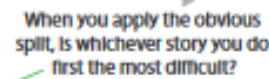
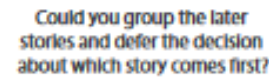
DEFER PERFORMANCE



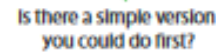
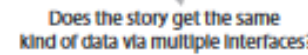
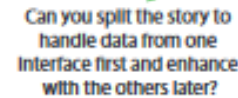
SIMPLE/COMPLEX



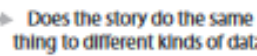
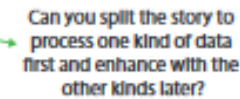
MAJOR EFFORT



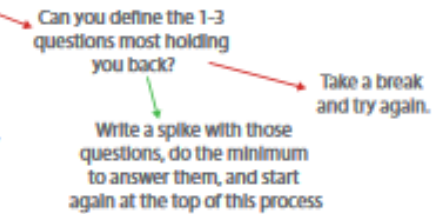
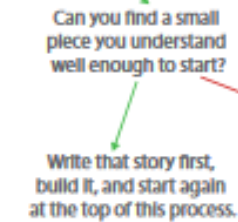
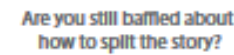
INTERFACE VARIATIONS



VARIATIONS IN DATA



BREAK OUT A SPIKE



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



AGILE FOR ALL
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.

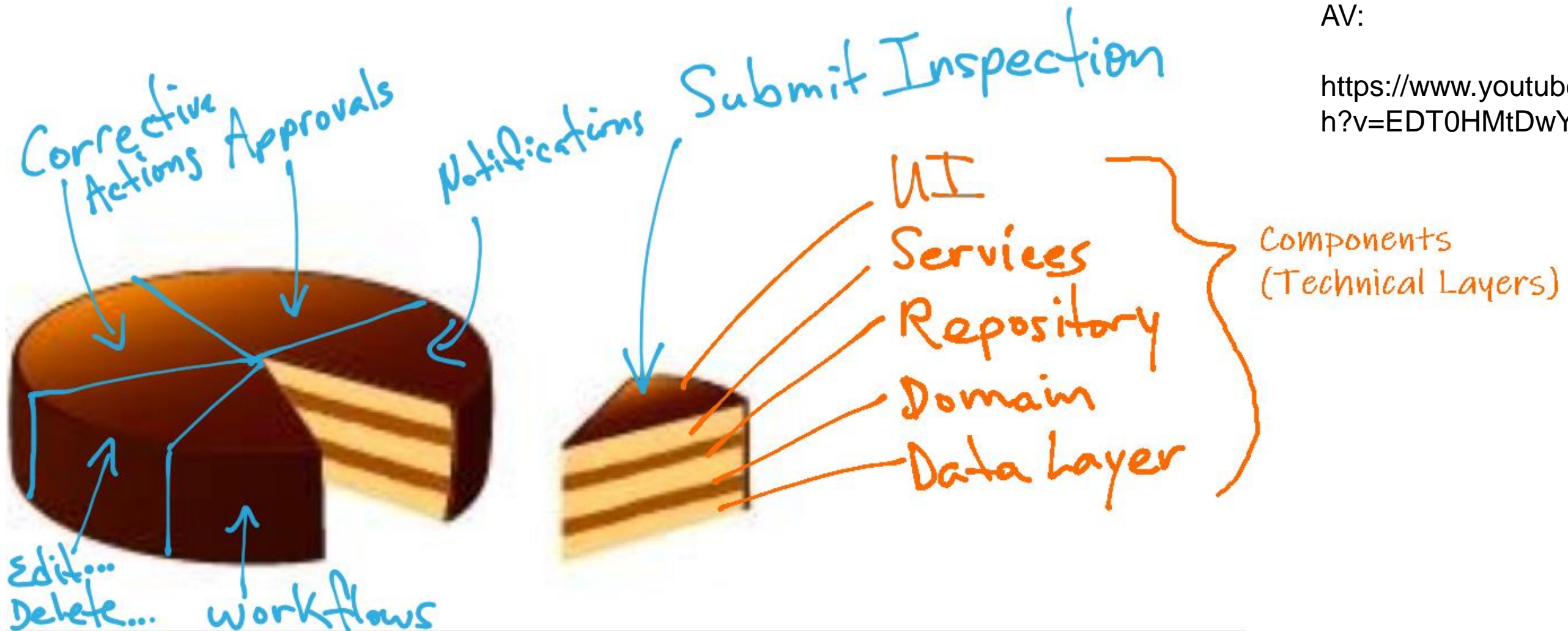
Last updated 2/21/2018

Vertical Slicing

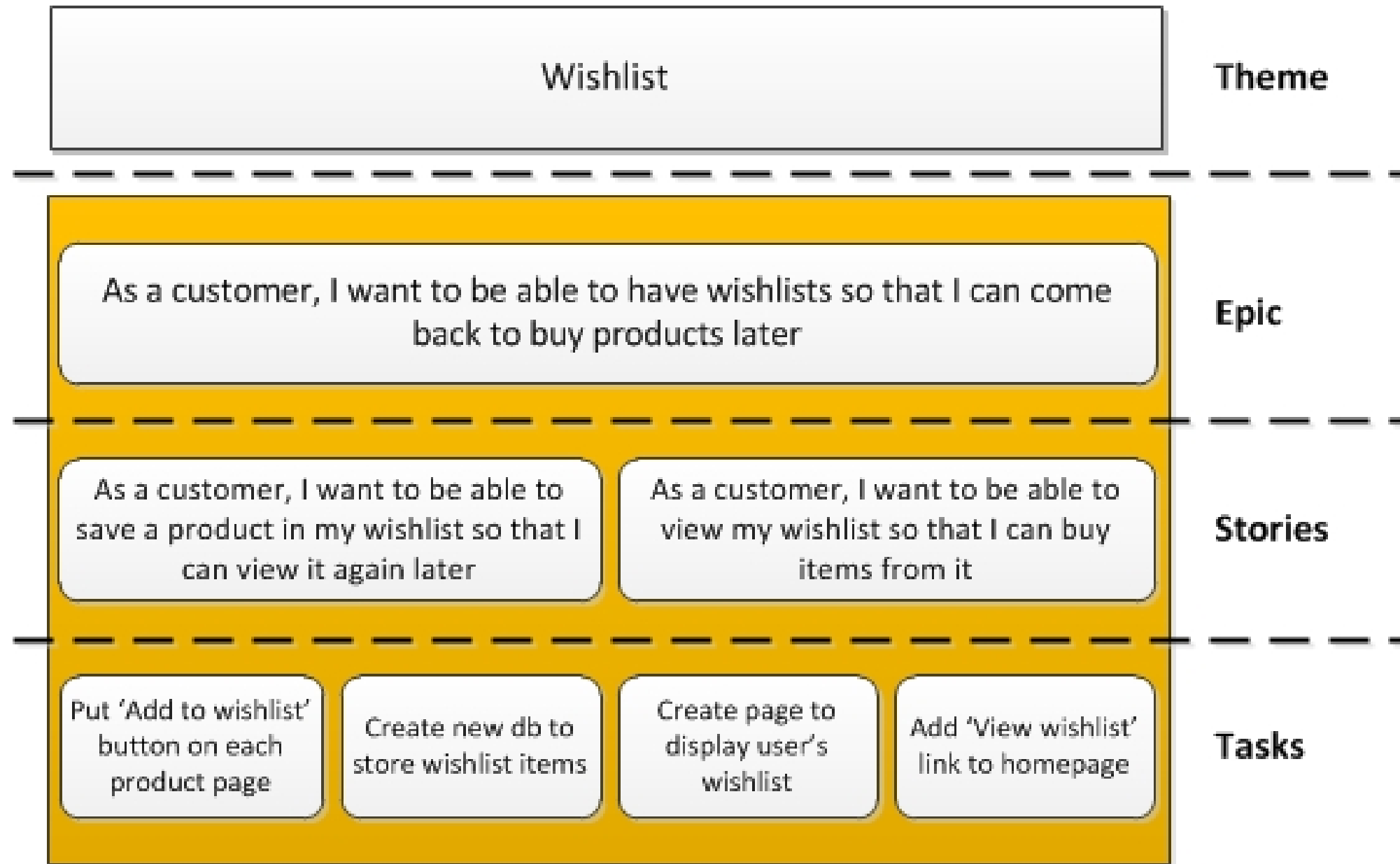
Work on "Slices" of a Cake VS Layers

AV:

<https://www.youtube.com/watch?v=EDT0HMtDwYI>

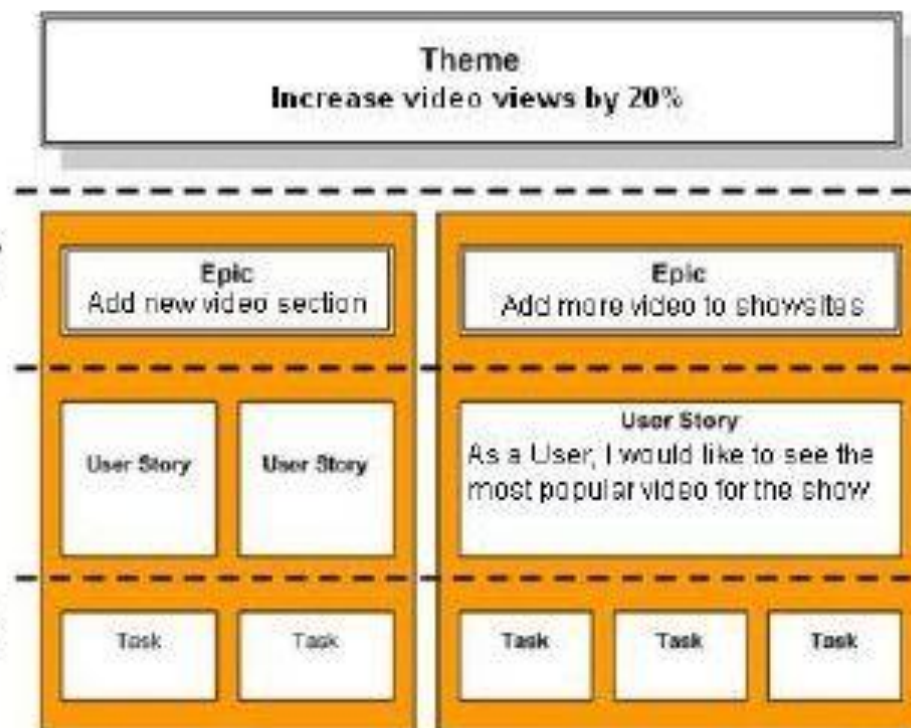


Theme Epic Story Tasks



Themes, Epics and Stories

- **Theme** or “tent-pole”: a top-level objective or vision.
- **Epic**: a group of related Stories that describes a particular higher level capability or functionality.
- **Story***: an *Independent, Negotiable, Valuable, Estimatable, Small, Testable* candidate requirement (“INVEST”).
- **Task**: Stories will often be broken down into specific work sub-tasks.



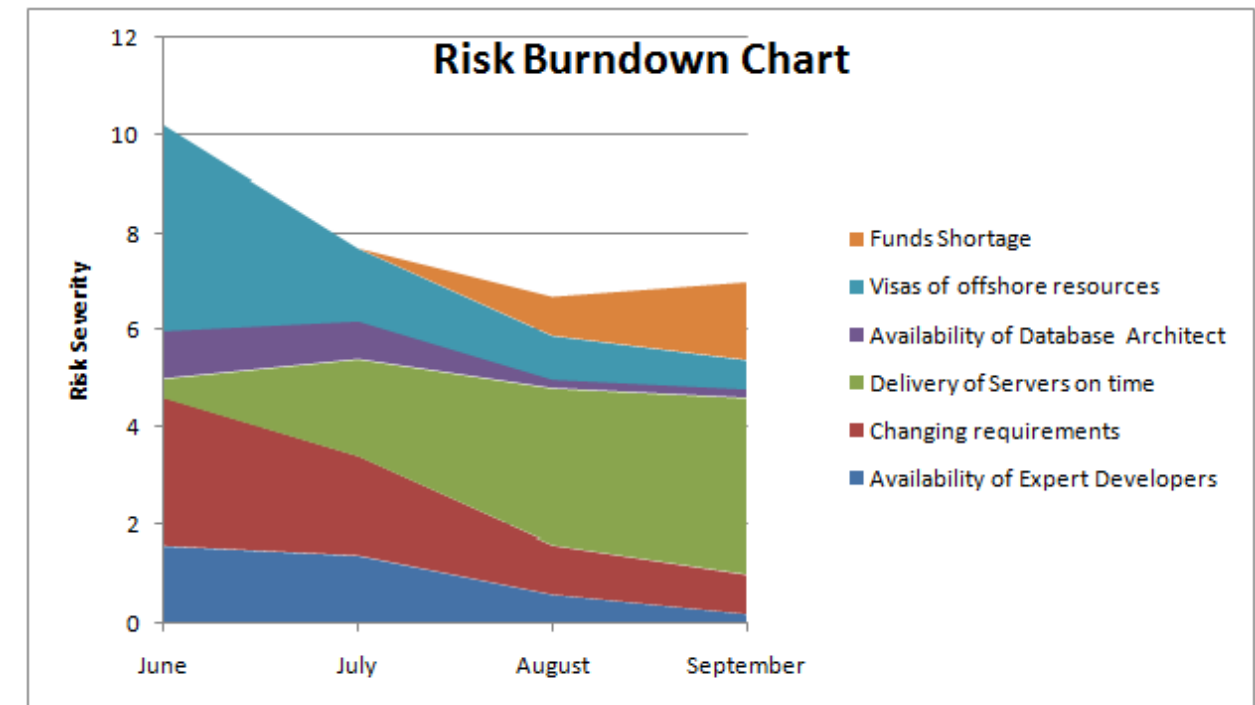
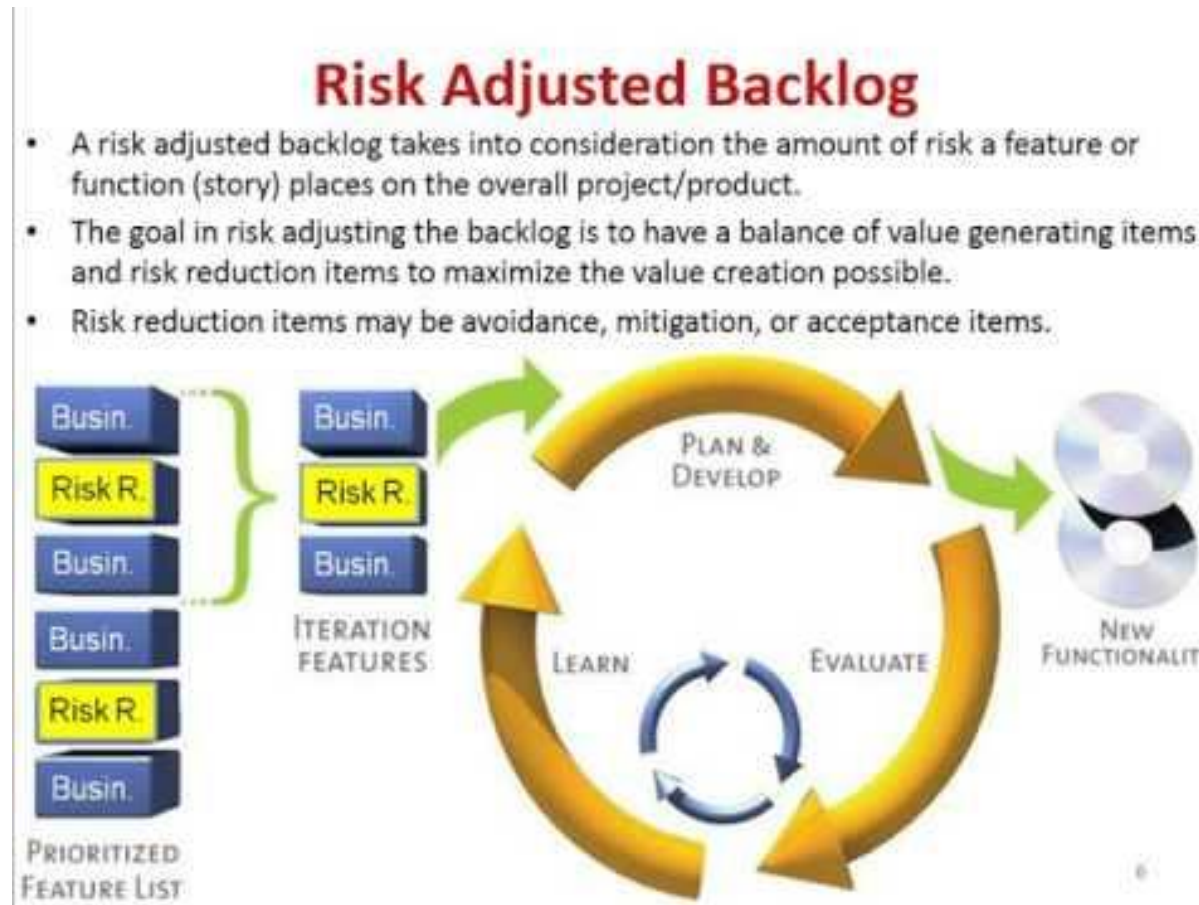
The **Product Manager** is responsible for making sure that work is broken down appropriately at each stage. For example, Themes and even Epics are okay during Portfolio reviews but all Epics should be broken down to the Story level once Grooming meetings are complete.

Risk adjusted backlog



Including but not limited to:

- risk adjusted backlog
- risk burn down graphs
- risk-based spike
- architectural spike



Product Backlog is DEEP; INVEST Wisely and DIVE Carefully



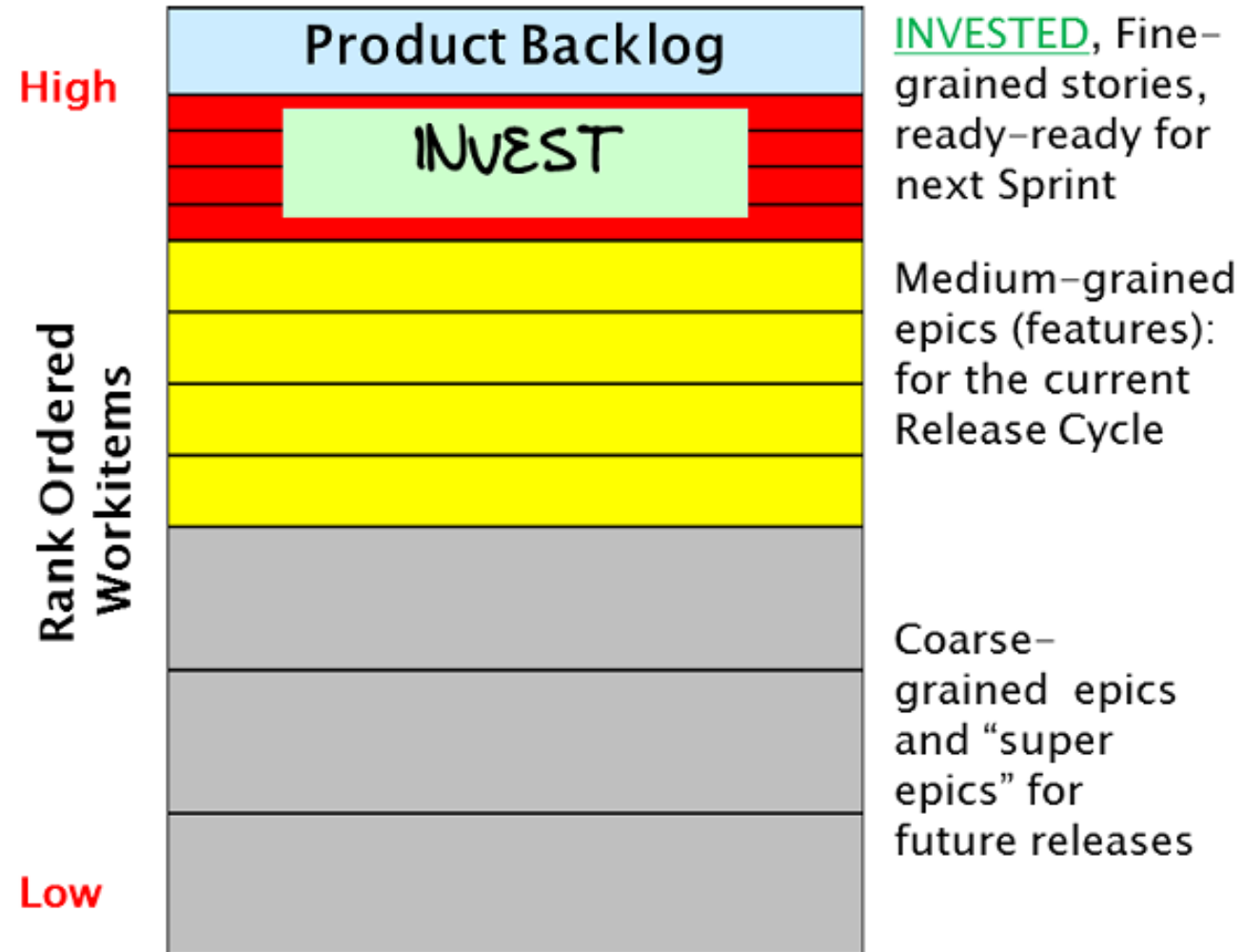
- Detailed appropriately
- Estimated appropriately
- Emergent
- Prioritized as needed

INVEST in stories for the next sprint

- Independent
- Negotiable
- Valuable
- Estimable
- Sized appropriately
- Testable

Product backlog workitems are **linearly ordered** based on the DIVE criteria:

- Dependencies
- Insure against Risks: Business and technical
- Business Value
- Estimated Effort



Persona



- Drive design decisions based on the user interests
- Experience tailored

<https://www.behance.net/gallery/69055845/Persona-User-Journey-Map-Wireframes-E-Commerce>

<https://youtu.be/B23iWg0koi8>



Laura Miller

"The Creative Problem Solver"

Profile:

AGE 28
GENDER Female
STATUS Single (Engaged)
LIVES Pasadena, California
EDUCATION B.S. in Graphic Design at University of California, Berkeley
WORK Digital Graphic Design Manager at Guess, Inc.
INTEREST Shopping, Photography, Travelling, Yoga, Beauty

Personality:

Outgoing

Open-Minded

Empathetic

Creative

Impulsive

Adventurous

Technical Skills:

Microsoft Office Suite

Adobe Creative Cloud Apps

Social Media and Blogging

Personal Goals:

- Instruct Yoga classes to small groups
- Travel abroad and learn a new culture 1x per year
- Cook healthier meals for overall better health
- To become my own boss and start a business
- Read more healthy-living books

Likes & Dislikes:

LIKES

- To-do-lists
- Solving problems
- Social events
- Adventurous Activities

DISLIKES

- Procrastination
- Sales calls
- Insects

Favorite Brands:





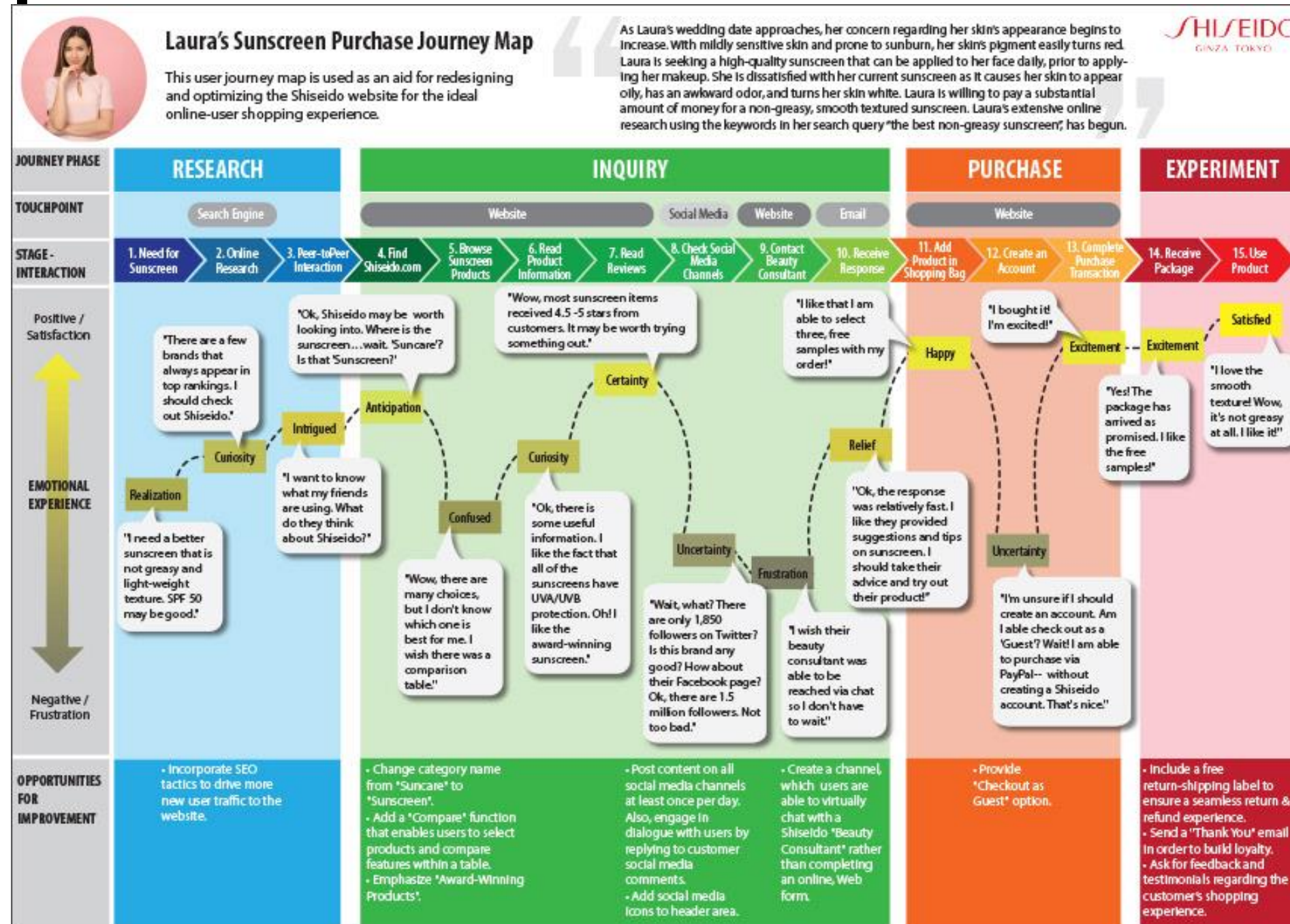

Bio:

Since graduating from university, Laura works as a Graphic Designer at the Guess's corporate office located in Los Angeles, CA. After six years of dedication and strong work ethic, Laura recently has been promoted to a manager's position. A few hobbies which Laura enjoys in her spare time are participating in yoga classes as well as training to become a part-time yoga instructor. At home, Laura is passionate about cooking healthy food and learning new recipes to improve her physical health. Staying fit, regardless of age, is her life-long goal. After her recent engagement to her fiancée, has become a busy person while planning her wedding, scheduled on September 15th. As a newlywed couple, Laura and her husband will visit Japan for their honeymoon; a place where she has always wanted to visit.



Journey Map

- Customer experience visible and tangible
- It captures customers' needs, processes, and perceptions at each touchpoint.
- This visualization allows companies to recognize the consumer's thoughts, feelings, actions, and pain points.
- it compels users to consider every touchpoint while determining how to optimize the overall, customer experience

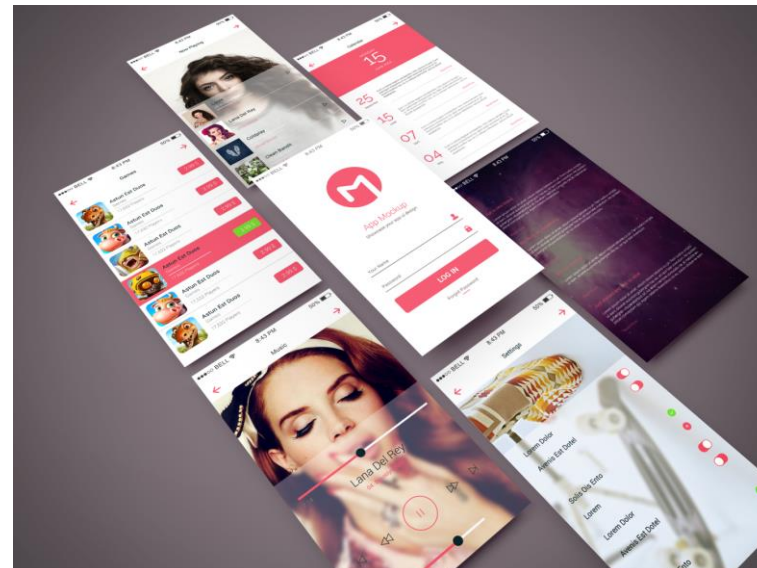


Wireframe



- It shows the main chunks of content
- It draws the outline and the layout structure
- It depicts the most basic UI
- Low fidelity
- Blueprint

Mockups



- Visual representation of final product look like
- Color scheme, visual, typography
- Mid/High fidelity
- Is not clickable

Prototype

- Representation of final product
- High fidelity
- Is clickable
- Interactive
- Provide user experience
- Upon UI approval, final product coding would start



<https://uxplanet.org/wireframe-mockup-prototype-what-is-what-8cf2966e5a8b>

<https://dribbble.com/search/app%20mockups>

Value \$\$ Prioritization Techniques

What is ROI

- ROI stand for Return on Investment.
- Return on Investment (ROI) is a performance measure used to evaluate the efficiency of an investment.
- To calculate ROI, the benefit (or return) of an investment is divided by the cost of the investment.
- The return on investment formula
- $ROI = (\text{Current Value of Investment} - \text{Cost of Investment}) / \text{Cost of Investment} * 100$
- **$ROI = \frac{10000\$ - 8000\$}{8000\$} * 100 = 25\%$**

Techat all

Case Study - Development Cost

Overnight delivery of payroll



Role	Annual Salary	Fully Burdened Labor Cost	Burdened Cost per Iteration	Time on Project	Adjusted Cost Per Iteration
Product Owner	≈50,000	≈75,000	≈2,900	100%	≈2,900
Programmer	≈50,000	≈75,000	≈2,900	100%	≈2,900
Programmer	≈30,000	≈45,000	≈1,700	50%	≈850
Analyst	≈40,000	≈60,000	≈2,300	100%	≈2,300
Tester	≈30,000	≈45,000	≈1,700	100%	≈1,700
Tester	≈50,000	≈75,000	≈2,900	100%	≈2,900
				Total	≈13,550

Measure	Cost
Cost Per Story Point	≈675
Cost Per Week	≈6,750
Cost Per Iteration	≈13,500

Cashflow



Table 10.9 Projected the returns from the WebPayroll project. (in thousands).

Quarter	Development Cost	New Revenue	Incremental Revenue	Retained Revenue	Operational Efficiencies	Net Cash Flow
1	-₹87,750	0	0	₹2,000	0	-₹85,750
2	-₹20,250	₹2,500	₹1,600	₹2,000	0	-₹14,150
3		₹3,750	₹5,000	₹2,000	₹7,500	₹18,250
4		₹3,750	₹7,500	₹2,000	₹7,500	₹20,750
5		₹7,500	₹10,000	₹4,000	₹7,500	₹29,000
6		₹7,500	₹10,000	₹4,000	₹7,500	₹29,000
7		₹7,500	₹10,000	₹4,000	₹15,000	₹36,500
8		₹7,500	₹10,000	₹4,000	₹15,000	₹36,500

The overnight feature is expected to be finished in the eighth iteration, or after sixteen weeks. The first quarter will be thirteen of those weeks for a cost of ₹87,750 (13X6750). The second quarter will be another three weeks for a cost of ₹20,250.

NPV

Table 10.10 Determining the NPV for WebPayroll.

End of Quarter	Net Cash Flow	Present Value Factor (12% / year)	Present Value
1	-85,750	0.971	-83,252
2	-14,150	0.943	-13,338
3	18,250	0.915	16,701
4	20,750	0.888	18,436
5	29,000	0.863	25,016
6	29,000	0.837	24,287
7	36,500	0.813	29,677
8	36,500	0.789	28,813
		NPV (12%)= 46,341	

Net Present Value

The first formula we'll look at for evaluating a theme is the *net present value* (NPV). To determine NPV, sum the present values of each item in a stream of future values. The formula for doing so is:

$$NPV(i) = \sum_{t=0}^n F_t(1+i)^{-t}$$

where i is the interest rate and F_t is the net cash flow in period t .

Payback



Table 10.13 Determining the payback period for the WebPayroll overnight project.

Quarter	Net Cash Flow at End of Quarter	Running Total
1	-a85,750	-a85,750
2	-a14,150	-a99,900
3	a18,250	-a81,650
4	a20,750	-a60,900
5	a29,000	-a31,900
6	a29,000	-a2,900
7	a36,500	a33,600
8	a36,500	a70,100

Discounted Payback



Table 10.14 Determining the WebPayroll overnight project's discounted payback period.

End of Quarter	Net Cash Flow	Present Value Factor (12%/year)	Discounted Cash Flow	Running Total
1	-85,750	0.971	-83,252	-83,252
2	-14,150	0.943	-13,338	-96,590
3	18,250	0.915	16,701	-79,889
4	20,750	0.888	18,436	-61,453
5	29,000	0.863	25,016	-36,437
6	29,000	0.837	24,287	-12,150
7	36,500	0.813	29,677	17,527
8	36,500	0.789	28,813	46,340

Compare and Prioritize



Table 10.15 Various valuations for each theme in a project.

Theme	Story Points	Cost	NPV	ROI	Discounted Payback Period
Overnight service	150	≈101,250	≈46,341	45%	7 quarters
Custom reporting	90	≈60,750	≈34,533	15%	6 quarters
Partner integration	60	≈40,500	≈30,013	49%	3 quarters

NPV Vs IRR

Table 10.11 Comparing two projects across NPV and IRR.

Project	Investment	NPV	IRR
Project A	α200,000	α98,682	27%
Project B	α100,000	α79,154	43%

Table 10.12 Cash flows for the projects in Table 10.11.

Year	Project A	Project B
0	-200,000	-100,000
1	50,000	50,000
2	75,000	75,000
3	100,000	50,000
4	170,000	50,000

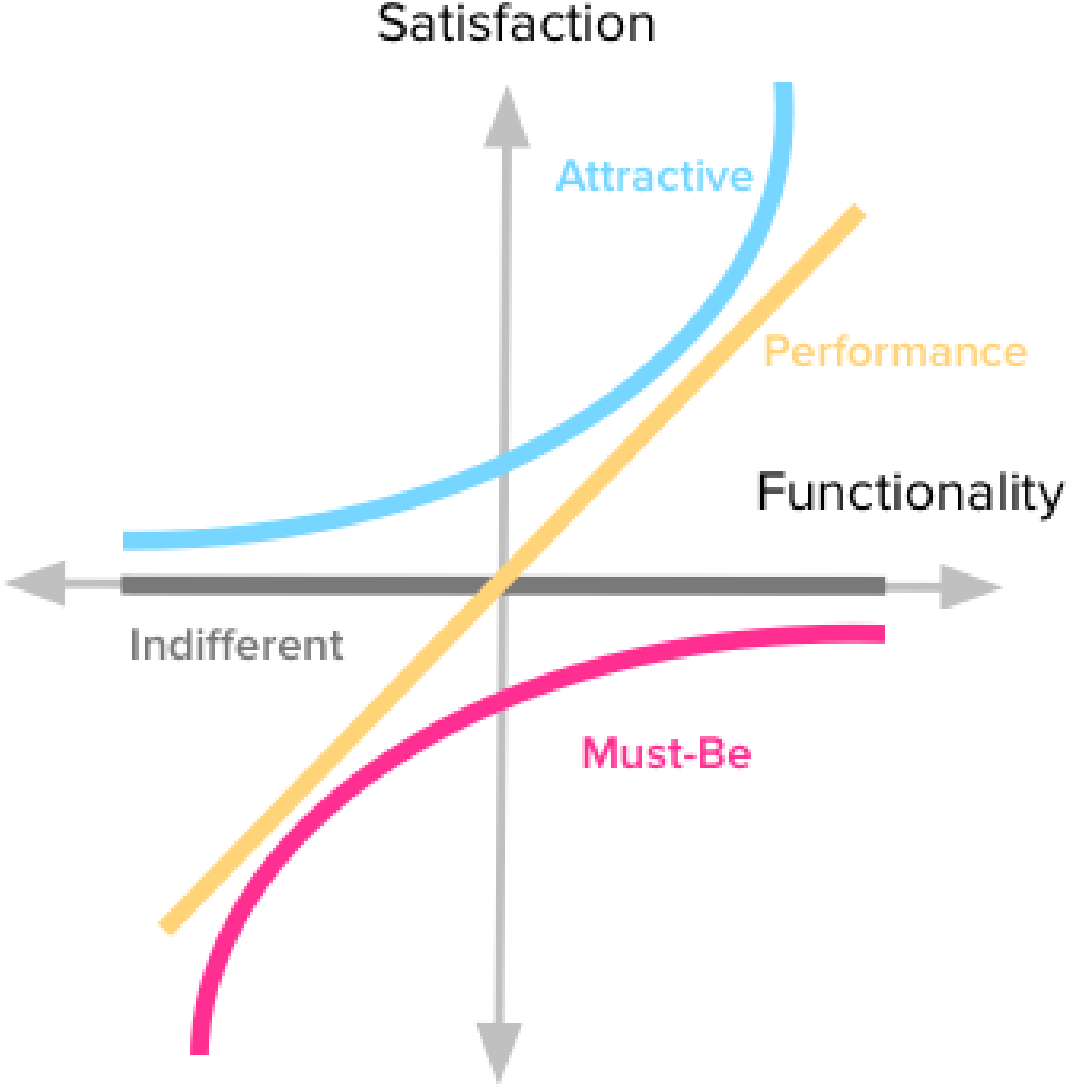
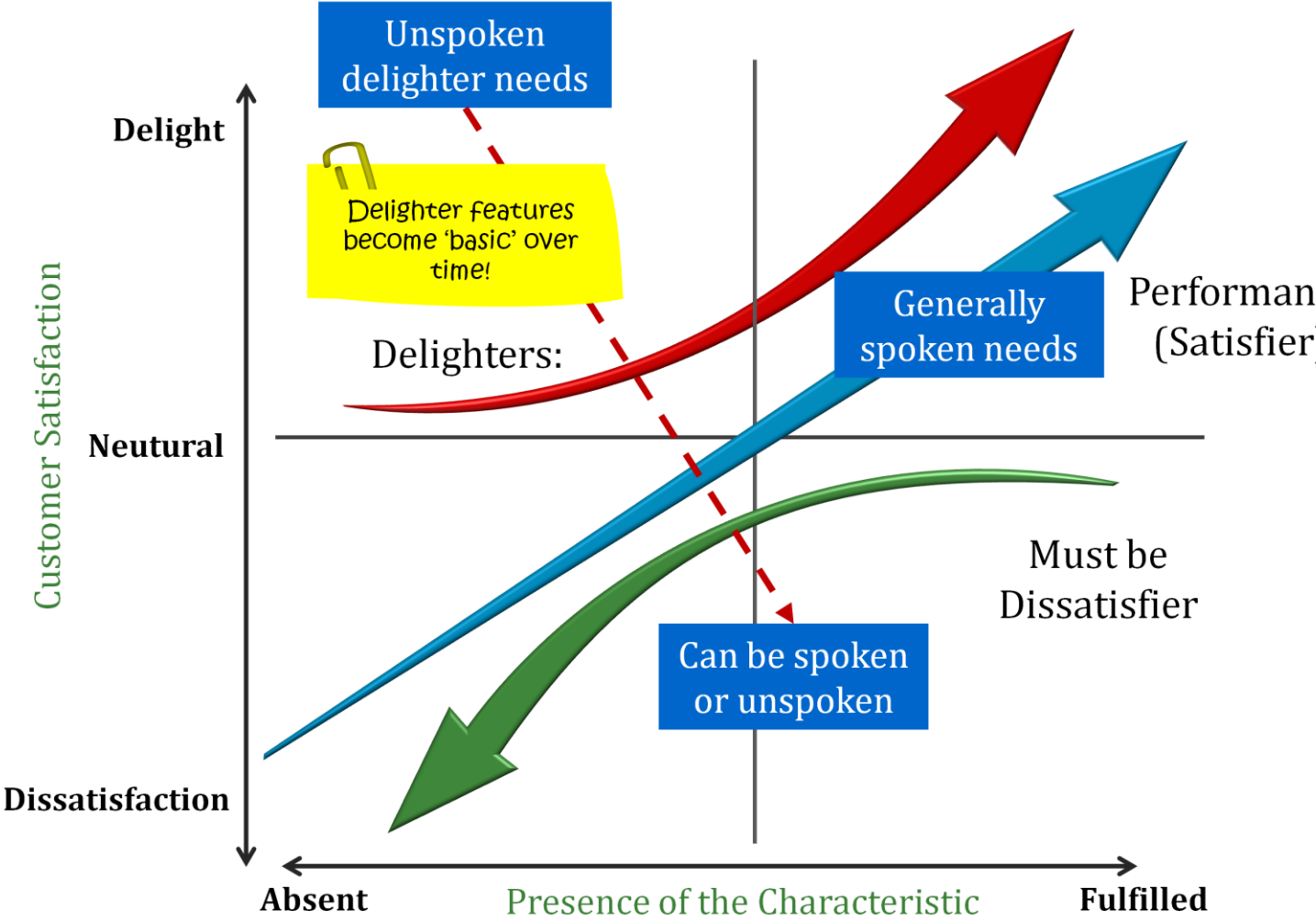
MoSCoW



<https://www.youtube.com/watch?v=QfZo9cxnQgY>



Kano Analysis



As a general rule of thumb, features should be prioritized such that this order is followed: Must-Be > Performance > Attractive > Indifferent.

Dysfunctional (feature absent)



Functional
(feature present)

	Like it	Expect it	Don't Care	Live With	Dislike
Like it	Q	A	A	A	P
Expect it	R	Q	I	I	M
Don't Care	R	I	I	I	M
Live With	R	I	I	Q	M
Dislike	R	R	R	R	Q

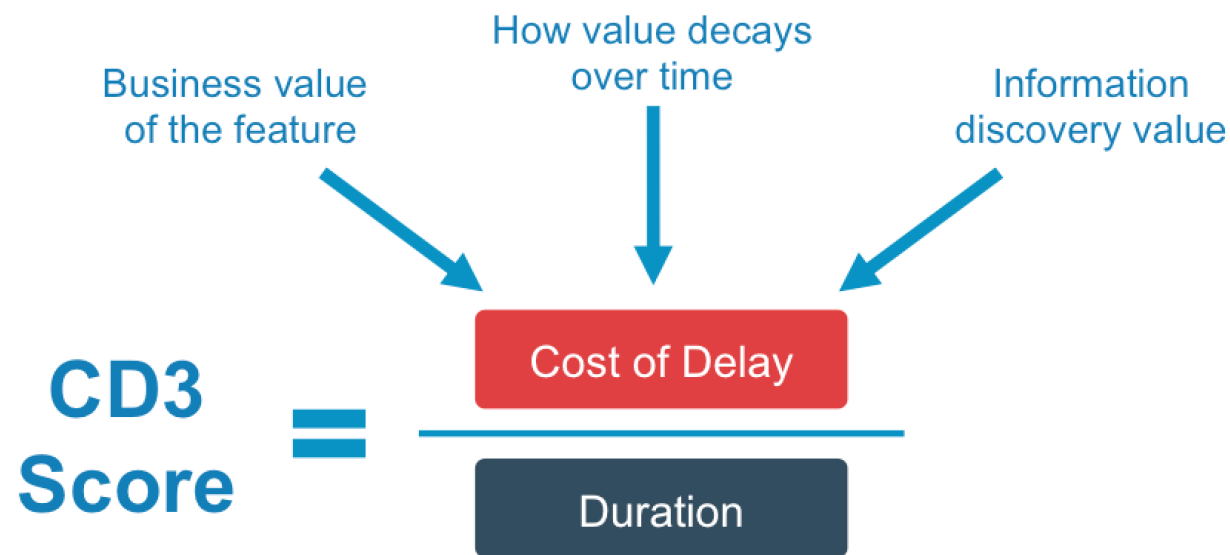
Feature	M	P	A	I	R	Q	Total	Category
Feature 1	9	2	1	1		2	15	M
Feature 2	2	11		2			15	P
...			
Feature N	2	2	8		3		15	A

CD3 Prioritization



<https://www.planview.com/resources/articles/lkdc-cost-delay/#:~:text=From%20there%2C%20you%20can%20calculate,divide%20the%20answer%20by%201%2C000.>

CD3:
Cost of Delay Divided by Duration



	Cost of Delay	Duration	CD3 Score
Feature A	\$1,000/week	5 weeks	200
Feature B	\$4,000/week	1 week	4,000
Feature C	\$5,000/week	2 weeks	2,500

CD3



Feature	Duration	Value	CD3
A	4 weeks	\$1,500	0.375
B	2 weeks	\$2,000	1
C	10 weeks	\$8,500	0.85
D	7 weeks	\$6,000	0.857

Feature	Duration	Value	Cost
B	2 weeks	\$2,000	$2 \times 2,000 = 4,000$
A	4 weeks	\$1,500	$(4 + 2) \times 1,500 = 9,000$
D	7 weeks	\$6,000	$(7 + 6) \times 6,000 = 78,000$
C	10 weeks	\$8,500	$(10 + 13) \times 8,500 = 195,500$

Feature	Duration	Value	Cost
C	10 weeks	\$8,500	$10 \times 8,500 = 85,000$
D	7 weeks	\$6,000	$(10+7) \times 6,000 = 102,000$
B	2 weeks	\$2,000	$(2+17) \times 2,000 = 38,000$
A	4 weeks	\$1,500	$(4+19) \times 1,500 = 34,500$

Feature	Duration	Value	CD3	Cost
B	2 weeks	\$2,000	1	$2 \times 2,000 = 4,000$
D	7 weeks	\$6,000	0.857	$(7 + 2) \times 6,000 = 54,000$
C	10 weeks	\$8,500	0.85	$(10 + 9) \times 8,500 = 161,500$
A	4 weeks	\$1,500	0.375	$(4 + 19) \times 1,500 = 34,500$

Prioritization Option	Cost of Delay
No Priority	\$414,000
Duration	\$286,500
Value	\$259,500
CD3	\$254,500

Value based prioritization

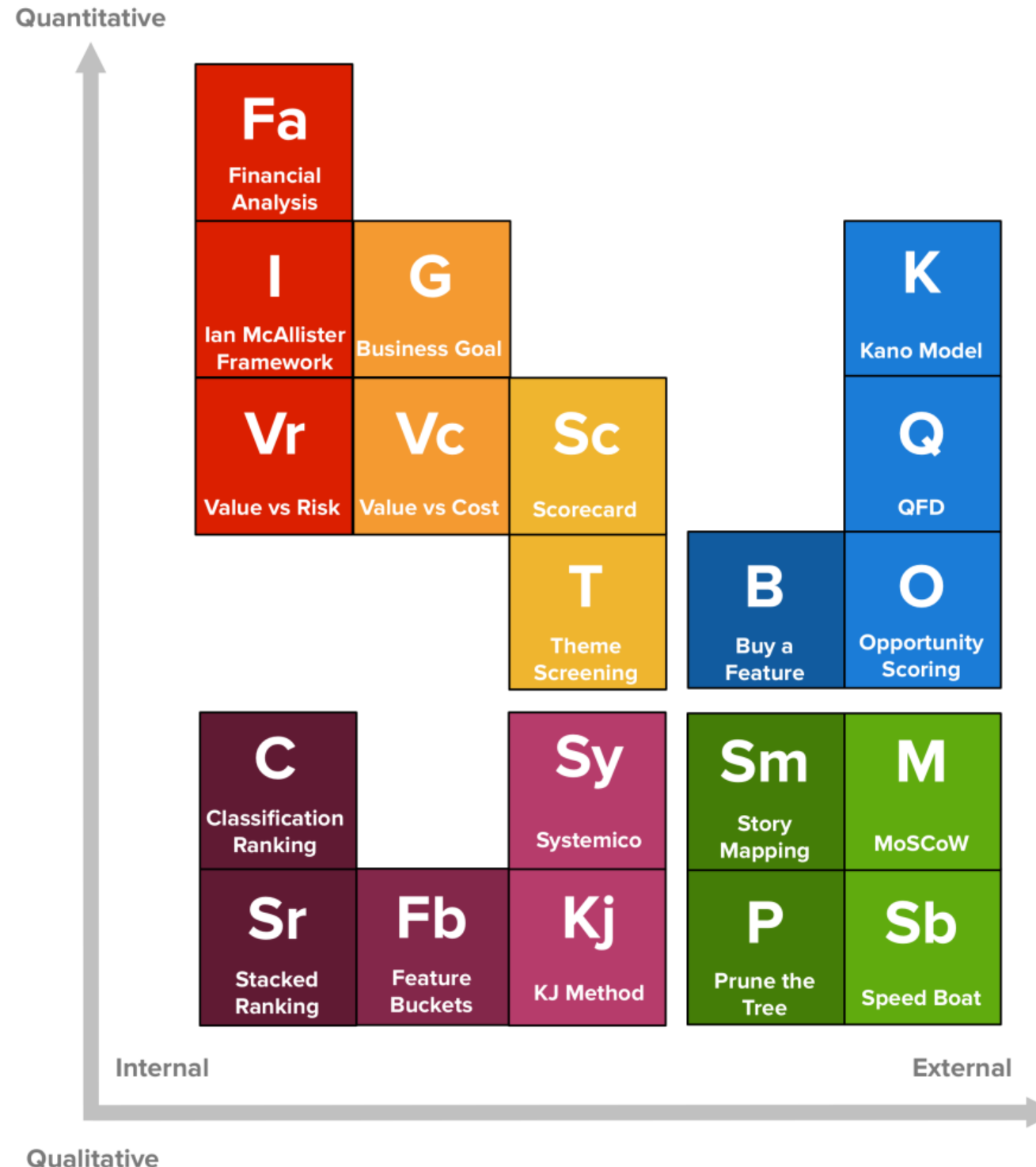


Including but not limited to:


- ROI/NPV/IRR
- compliance
- customer valued prioritization
- requirements reviews
- minimal viable product (MVP)
- minimal marketable feature (MMF)
- relative prioritization/ranking
- MoSCoW
- Kano analysis



Stakeholder Prioritization Techniques



<https://foldingburritos.com/product-prioritization-techniques/>

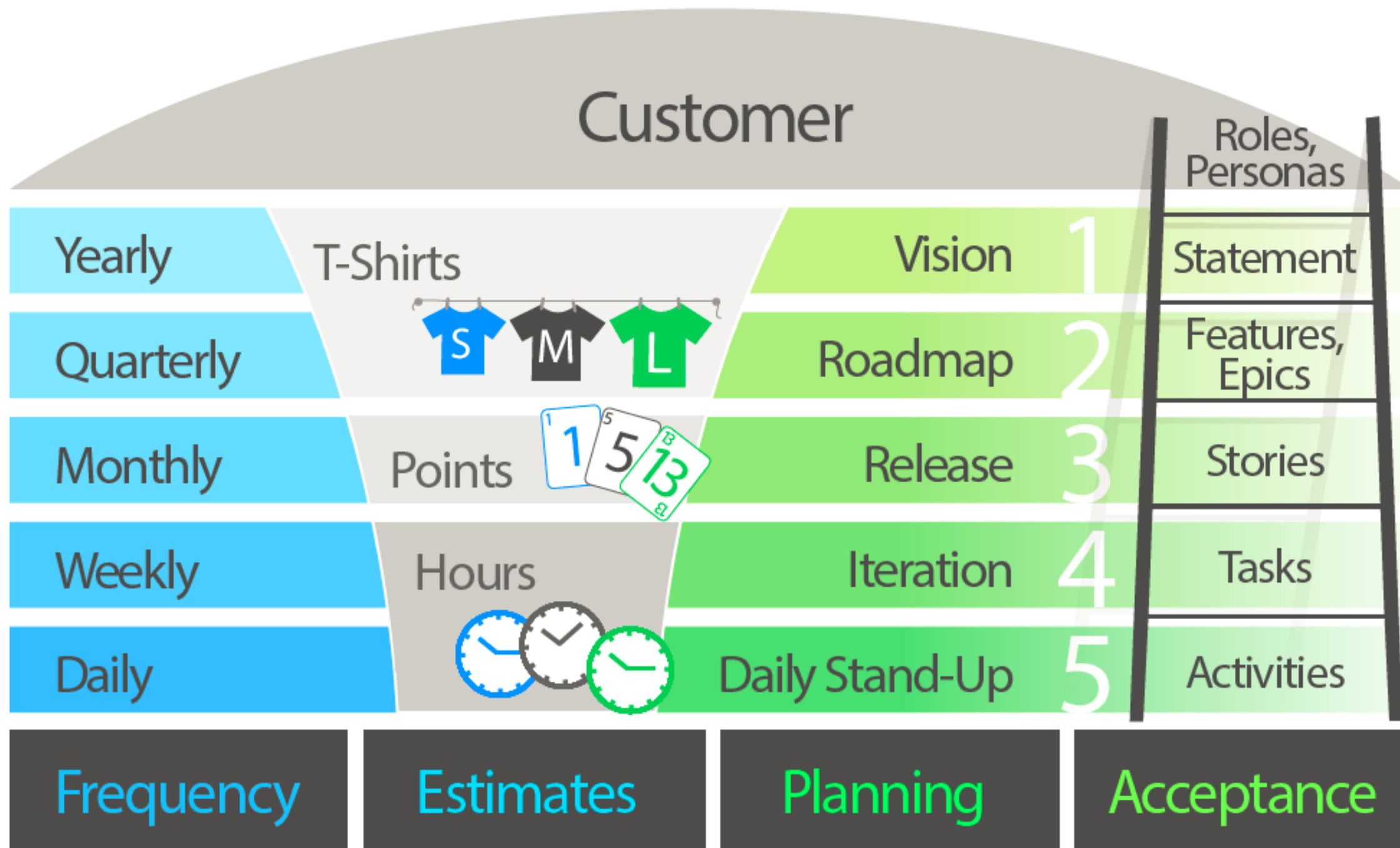


AGILE PLANNING & ESTIMATION

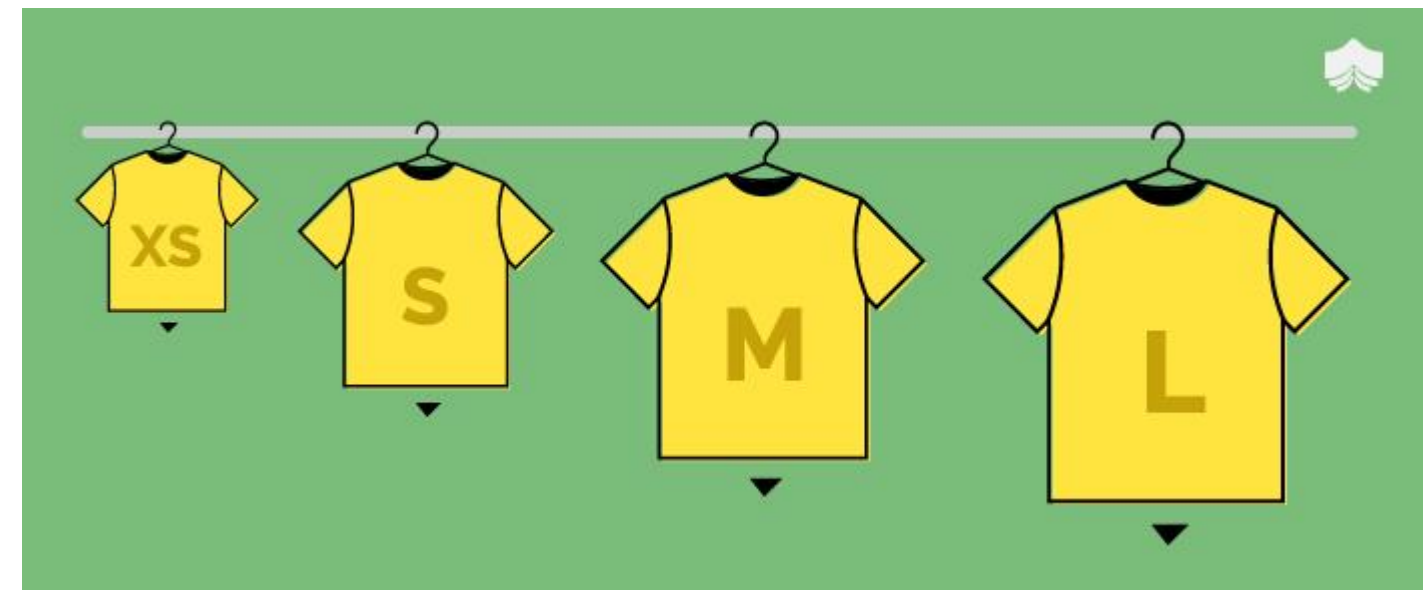
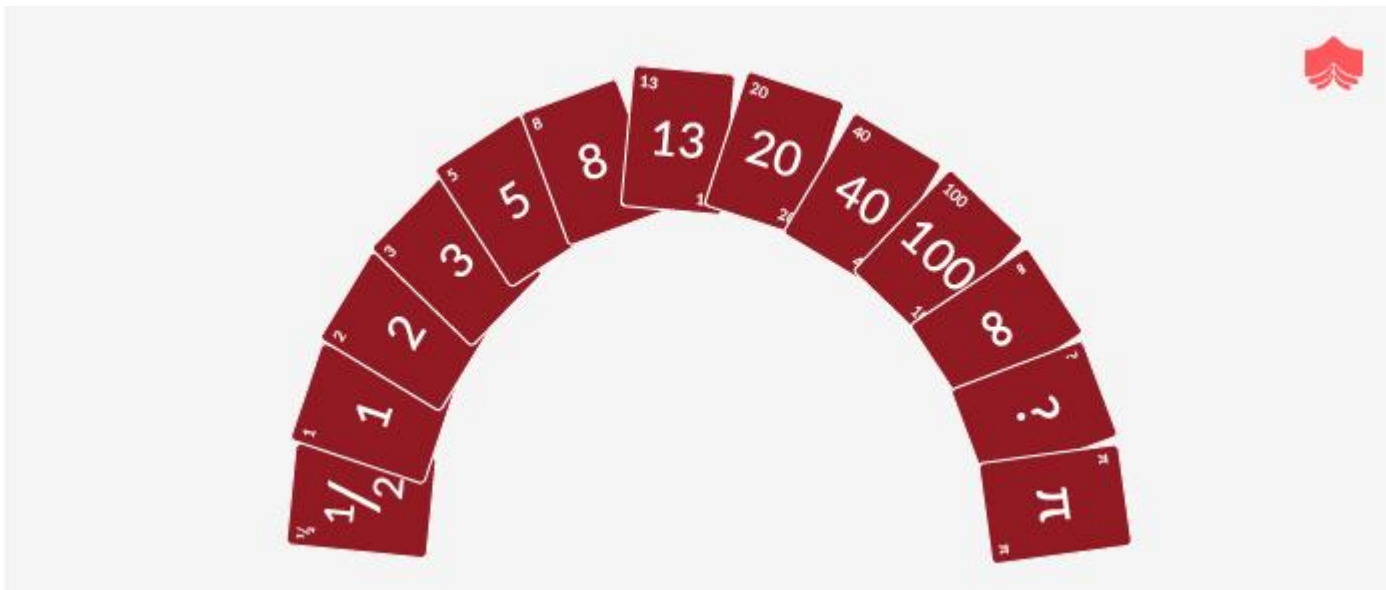
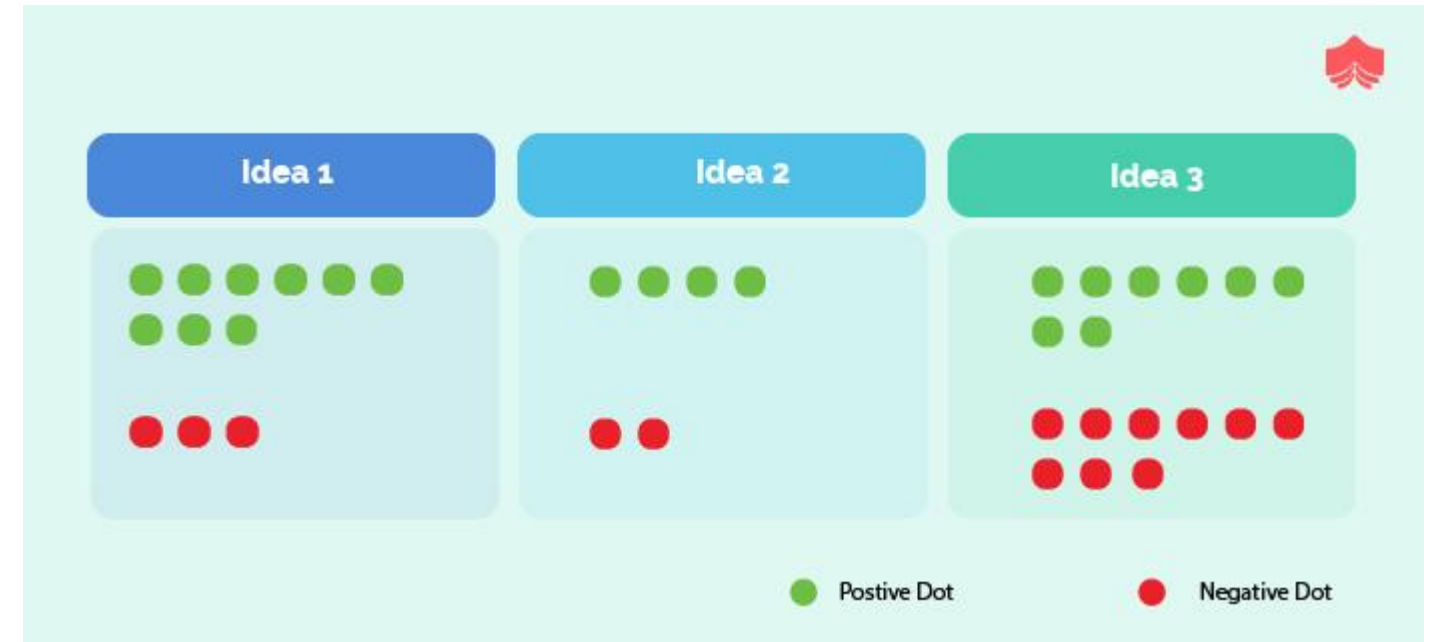
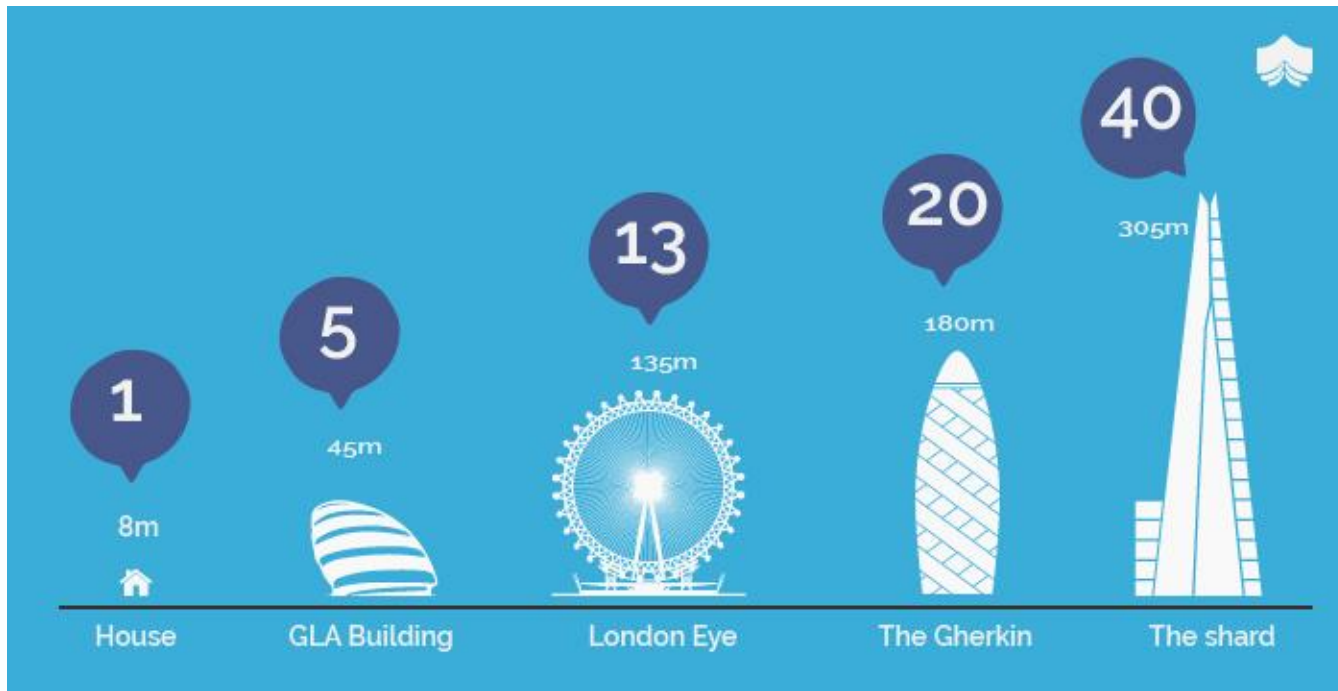
Agile Estimating

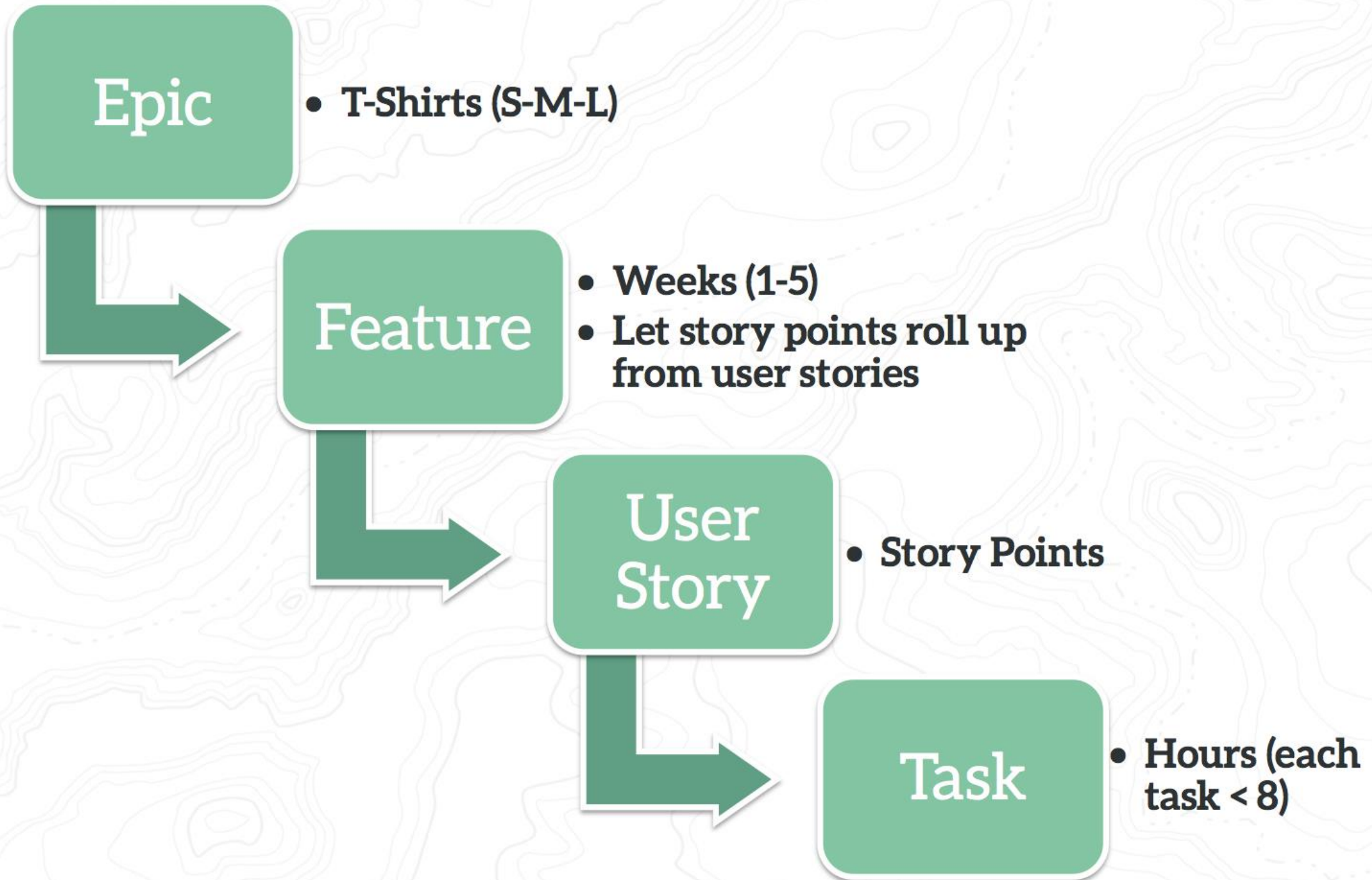


- Affinity Estimation - Relative Sizing - XS, S, M, L, XL – Product Backlog Stage
- Planning Poker – Iteration/Sprint Planning Stage
- Dot Voting
- T-Shirt sizing
- Ideal days

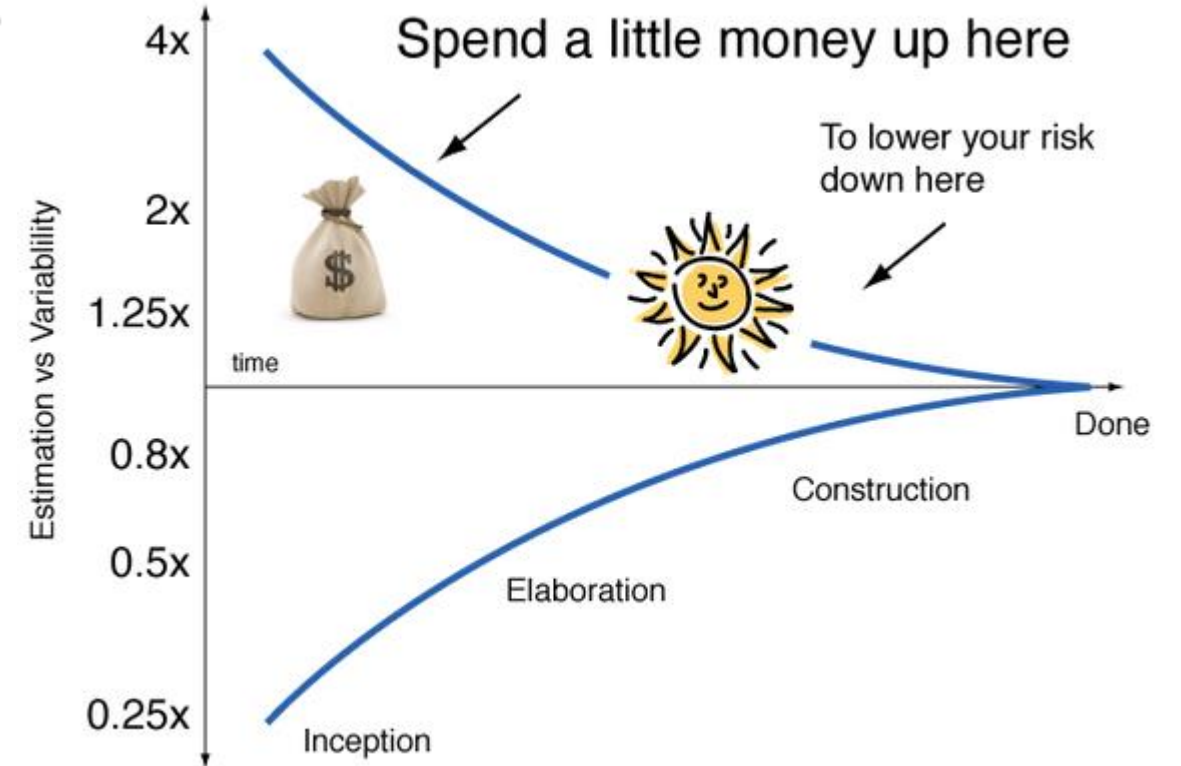
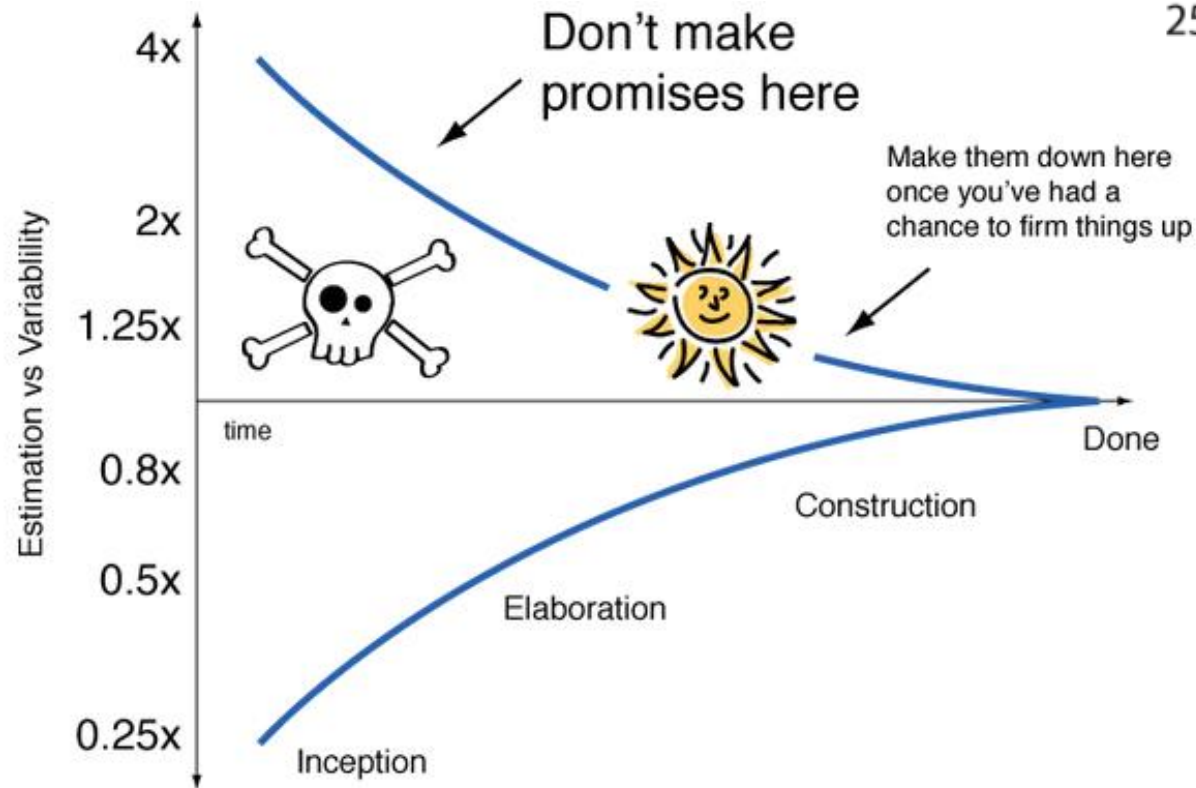
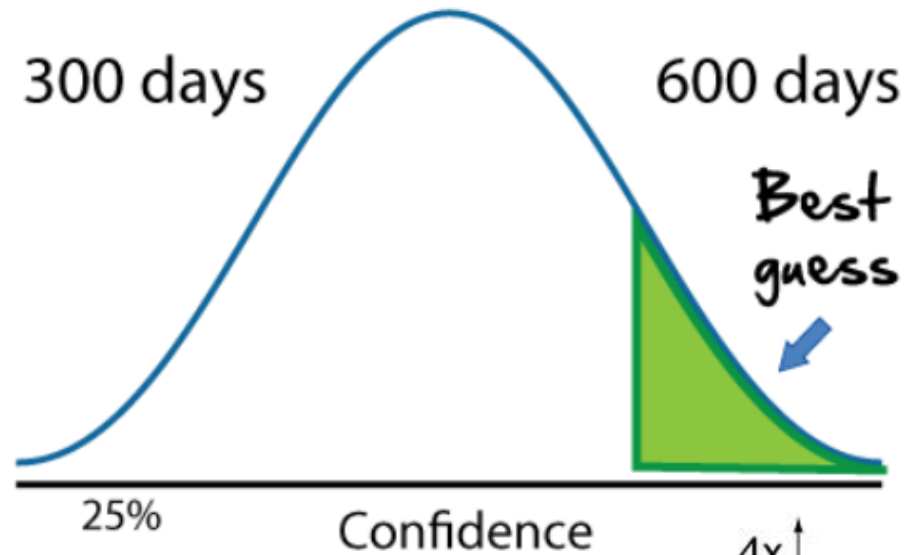


© Torak- www.torak.com





Cone of Uncertainty



AV on Estimation




“The primary purpose of software estimation is not to predict a project’s outcome; it is to determine whether a project’s targets are realistic enough to allow the project to be controlled to meet them.”

—Steve McConnell, *Software Estimation: Demystifying the Black Art*

<https://www.youtube.com/watch?v=sCCUEtjCpCs&feature=youtu.be> - Watch

Story Points Vs Ideal Days

Story Points vs. Ideal Days



<u>Story Points</u>	<u>Ideal Days (Real Time)</u>
<ul style="list-style-type: none">▶ Pros<ul style="list-style-type: none">- Pure measure of size- Requires true velocity project completion- Drive collaboration- Faster to estimate▶ Cons<ul style="list-style-type: none">- More abstract- People not use to them	<ul style="list-style-type: none">▶ Pros<ul style="list-style-type: none">- We have always done it this way- Executives want dates anyway▶ Cons<ul style="list-style-type: none">- Everyone's ideal (and real) day is different- Leads to unrealistic ideal schedule & commitments- More emotional process

In favour of story points...

1. Story points help drive cross-functional behaviour
2. Story point estimates don't decay
3. Story points are a pure measure of size
4. Estimating in story points is typically faster
5. My ideal days \neq your ideal days



Agile Planning Onion



5 levels of Agile Planning



Examples



For your benefit, here is the timing of the remaining development for everything that fits in the “inbound data” category Sasha refers to:

Epic Name	Epic ID	Priority	Sprint 10 5/13	Sprint 11 6/3	Sprint 12 6/24	Sprint 13 7/15	Sprint 14 8/5	Sprint 15 8/26
Asset-Based Fees	FBLT-78	Must Have	10	10	5			
Distribution Fees	FBLT-80	Must Have	DE Team	32	32	32		
Forfeiture Balance	FBLT-341	Must Have	DE Team	10	30			
IDA Fees	FBLT-82	Must Have			DE Team	30	30	

Planning, Monitoring and Adapting

Including but not limited to:
reviews

- Kanban board
- task board
- timeboxing
- iteration and release planning
- variance and trend analysis
- WIP limits
- daily stand ups
- burn down/up charts
- cumulative flow diagrams
- backlog grooming/refinement
- product-feedback loop



Different Agile Methods & Metrics

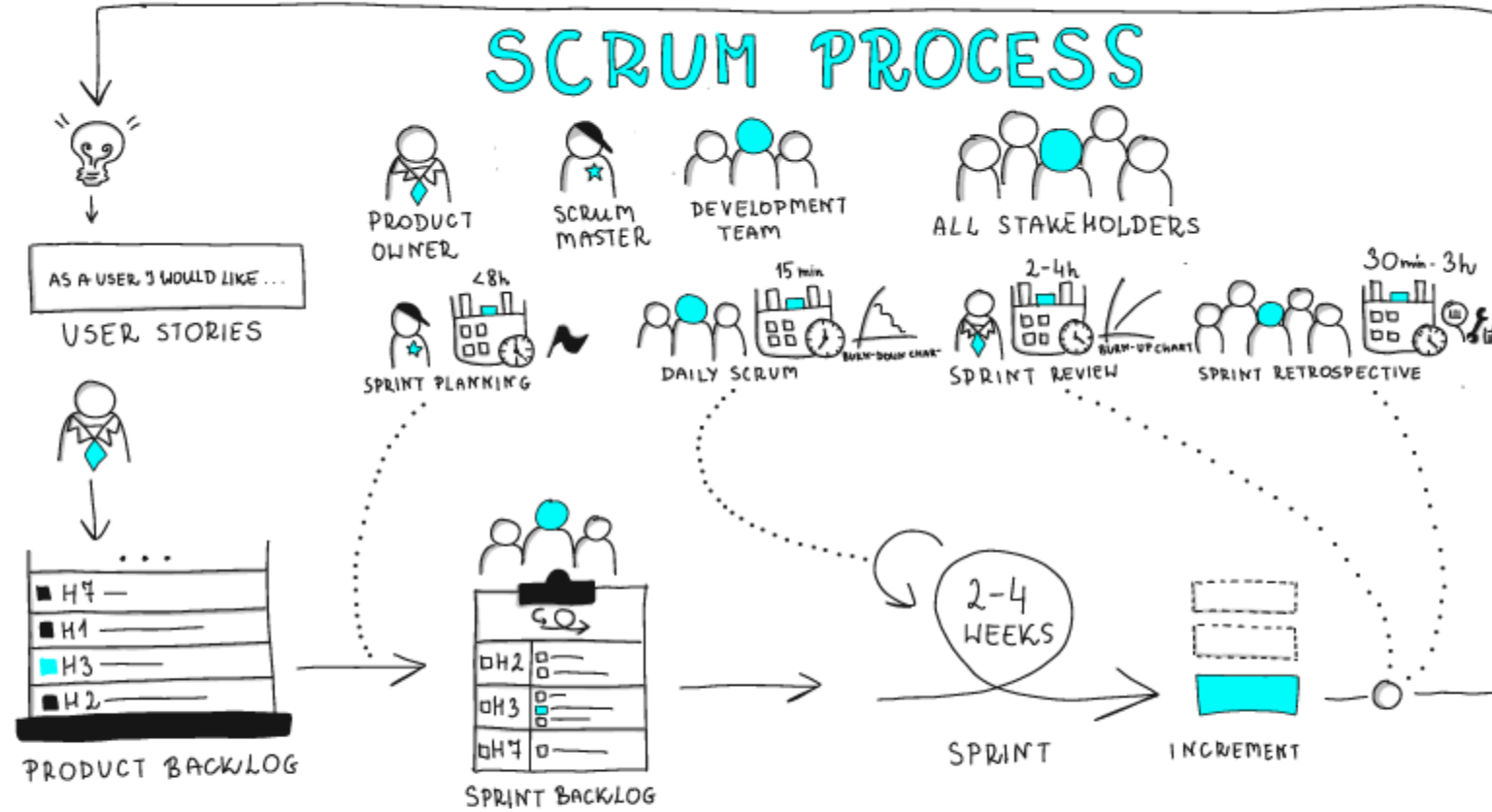
SCRUM

- Most used Agile methodology in the world

SCRUM 101

AGILE KNOWLEDGE BASE SKETCH #1

By Matt & Kat



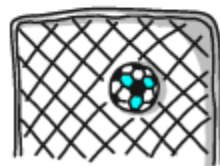
SCRUM VALUES

AGILE KNOWLEDGE BASE SKETCH #3

By Matt & Nat

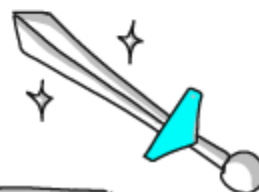
COMMITMENT

PEOPLE PERSONALLY COMMIT TO ACHIEVING THE GOALS OF THE SCRUM TEAM



COURAGE

SCRUM TEAM MEMBERS HAVE COURAGE TO DO THE RIGHT THING AND WORK ON TOUGH PROBLEMS



OPENNESS

THE SCRUM TEAM AND ITS STAKEHOLDERS AGREE TO BE OPEN ABOUT ALL THE WORK AND THE CHALLENGES WITH PERFORMING THE WORK



FOCUS

EVERYONE FOCUSES ON THE WORK OF THE SPRINT AND THE GOALS OF THE SCRUM TEAM



RESPECT

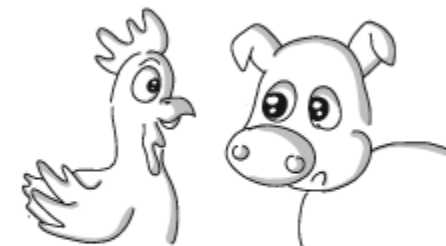
SCRUM TEAM MEMBERS RESPECT EACH OTHER TO BE CAPABLE, INDEPENDENT PEOPLE



© SketchingPM.com

HEY PIG, I WAS THINKIN' WE SHOULD OPEN A RESTAURANT.

I DON'T KNOW. WHAT WOULD WE CALL IT?



HOW ABOUT "HAM-N-EGGS"?



NO THANKS. I'D BE COMMITTED, BUT YOU'D ONLY BE INVOLVED!

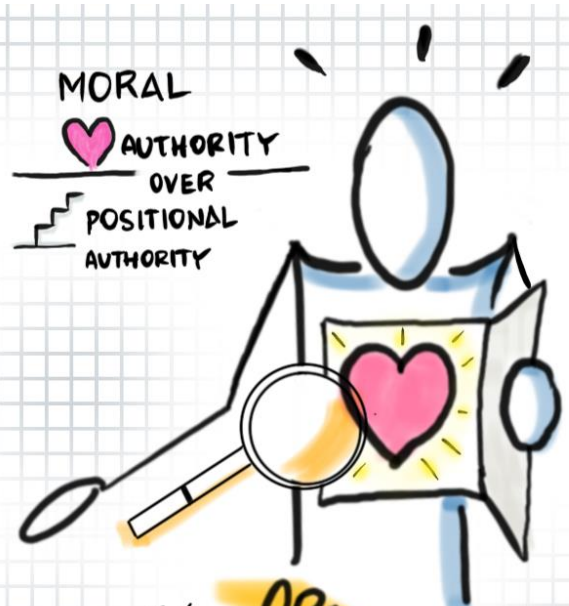


SCRUM Master

BY SERVING

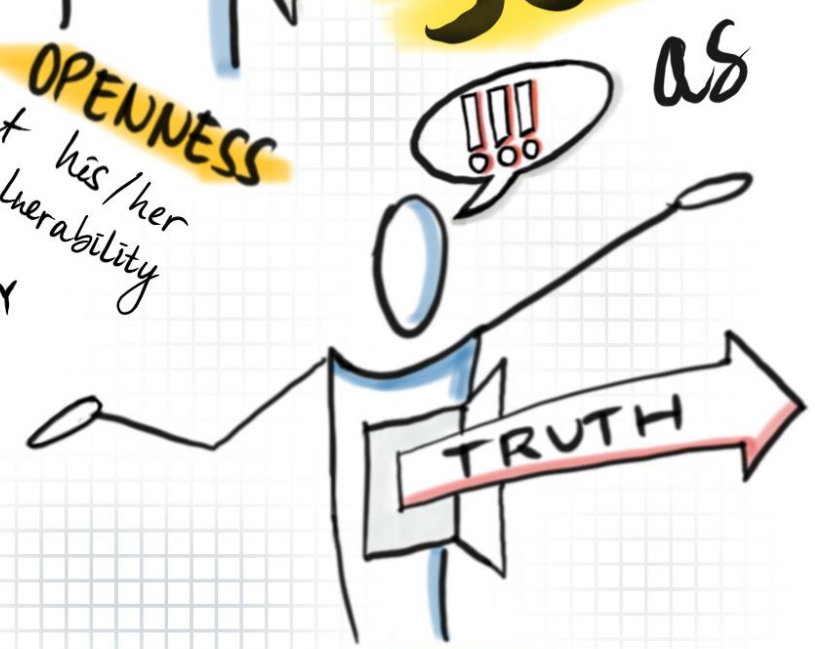
LEADING BY

as Servant Leader



OPENNESS
about his/her own vulnerability

LEADING BY BEING AN EXAMPLE



COURAGE
to tell the brutal truths

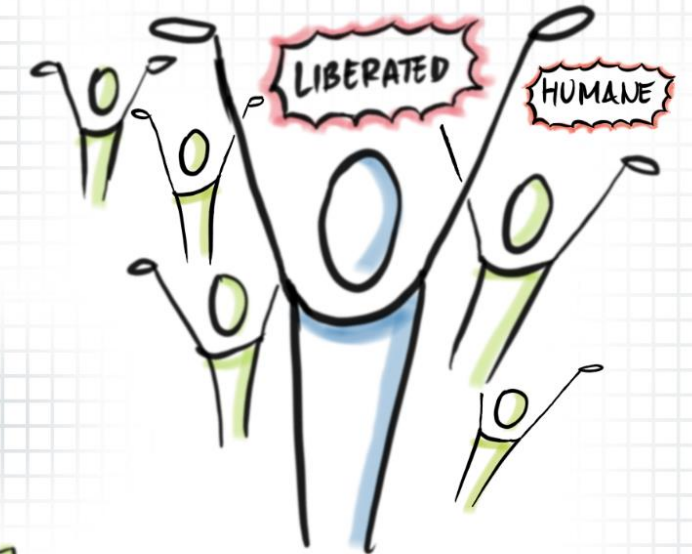


SELFLESSNESS
COMMITMENT
to put his/her needs last

LEADING WITH THE HEART



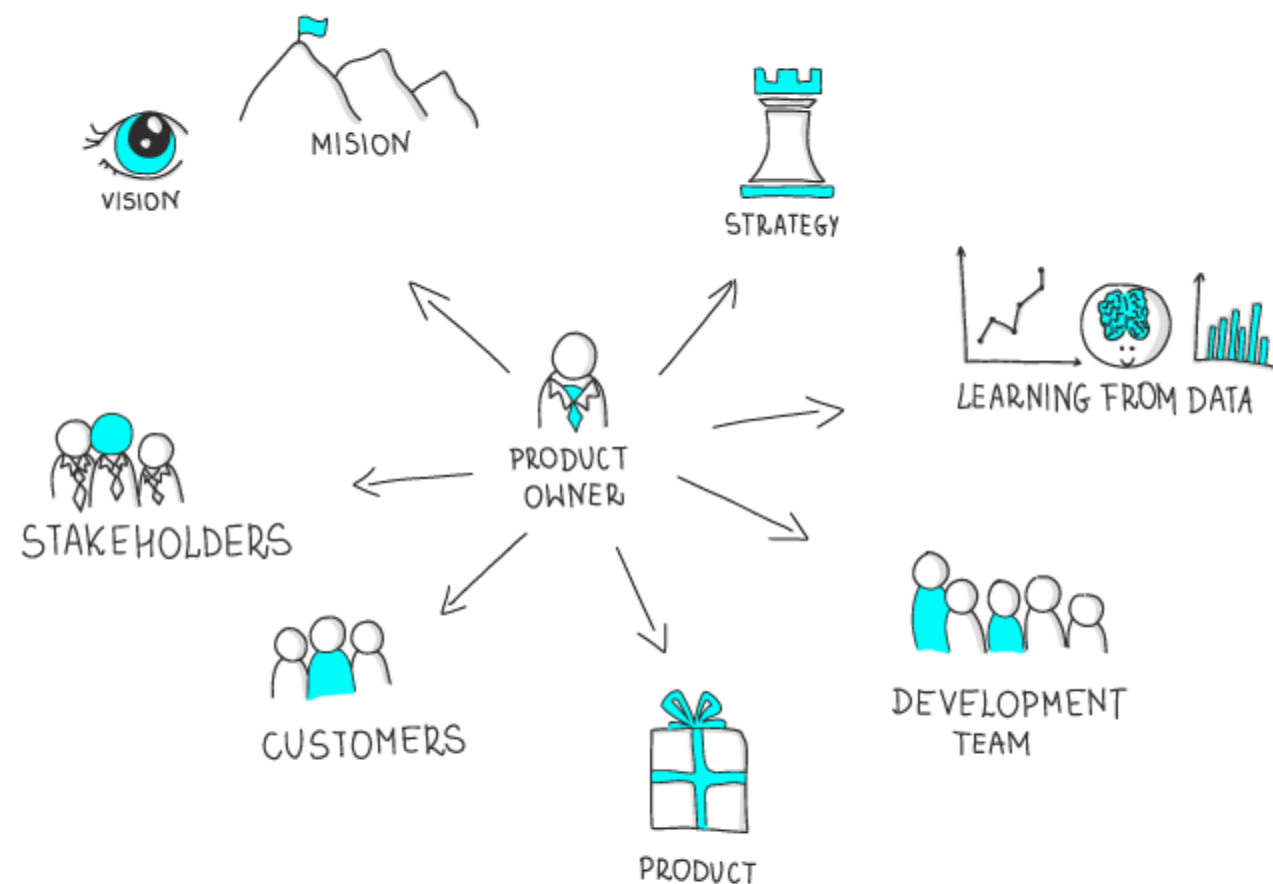
FOCUSED
on other's greatness



RESPECT
other's needs to be fully human

PRODUCT OWNER 101

AGILE KNOWLEDGE BASE SKETCH #8 *By Matt & Not*



* GOOD UNDERSTANDING OF MARKET RESEARCH METHODS & DATA ANALYTICS

* STRONG STAKEHOLDER MANAGEMENT SKILLS

* FULL-TIME ROLE & AVAILABLE

* PASSION FOR LEARNING & CONTINUOUS IMPROVEMENT

* EXCELLENT COMMUNICATION & PRESENTATION SKILLS

* STRONG BUSINESS & FINANCIAL ACUMEN

PO VS SM

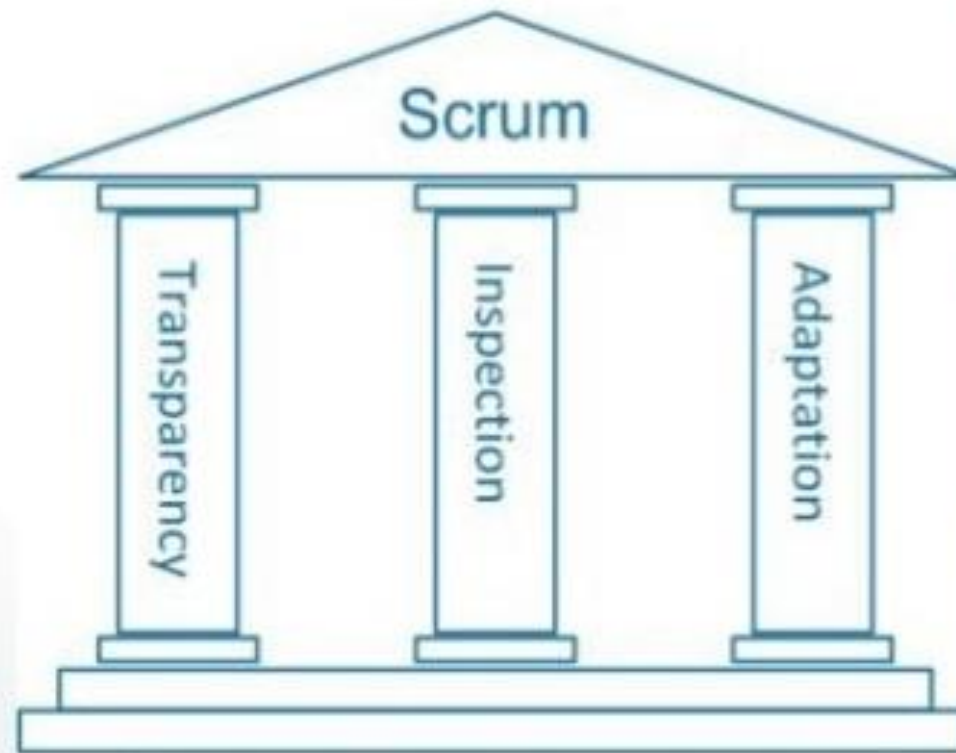
EVERY GREAT PRODUCT OWNER NEEDS A GREAT SCRUMMASTER (AND VICE VERSA)



Combining both roles – even partially – is not only very challenging but means that some duties are neglected.

Scrum – 3 Pillars

3 Pillars of Scrum

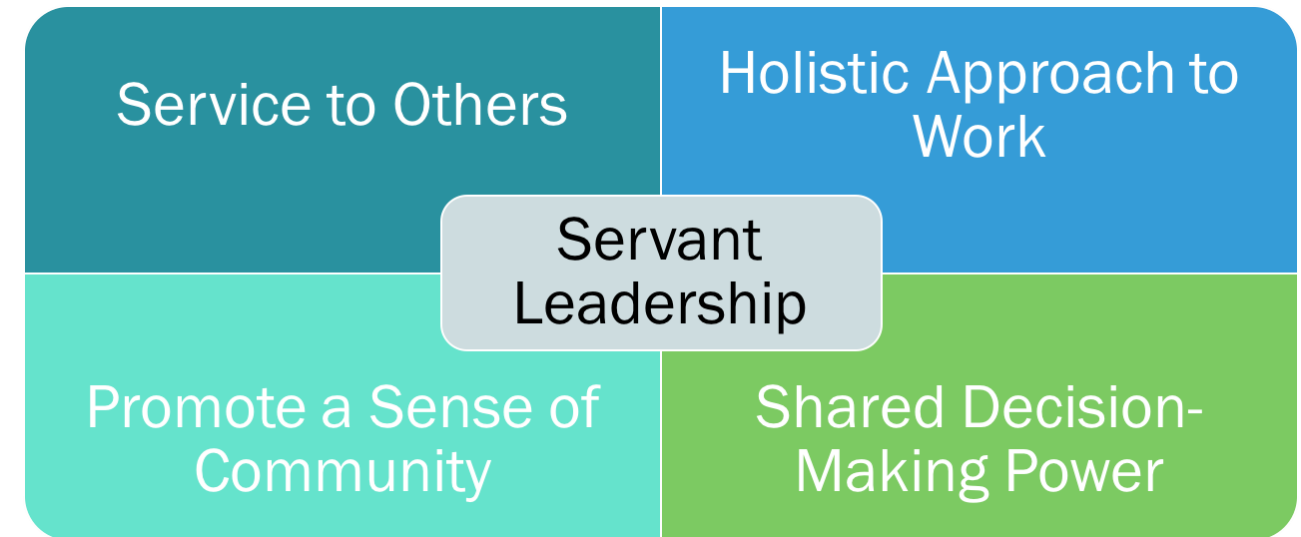
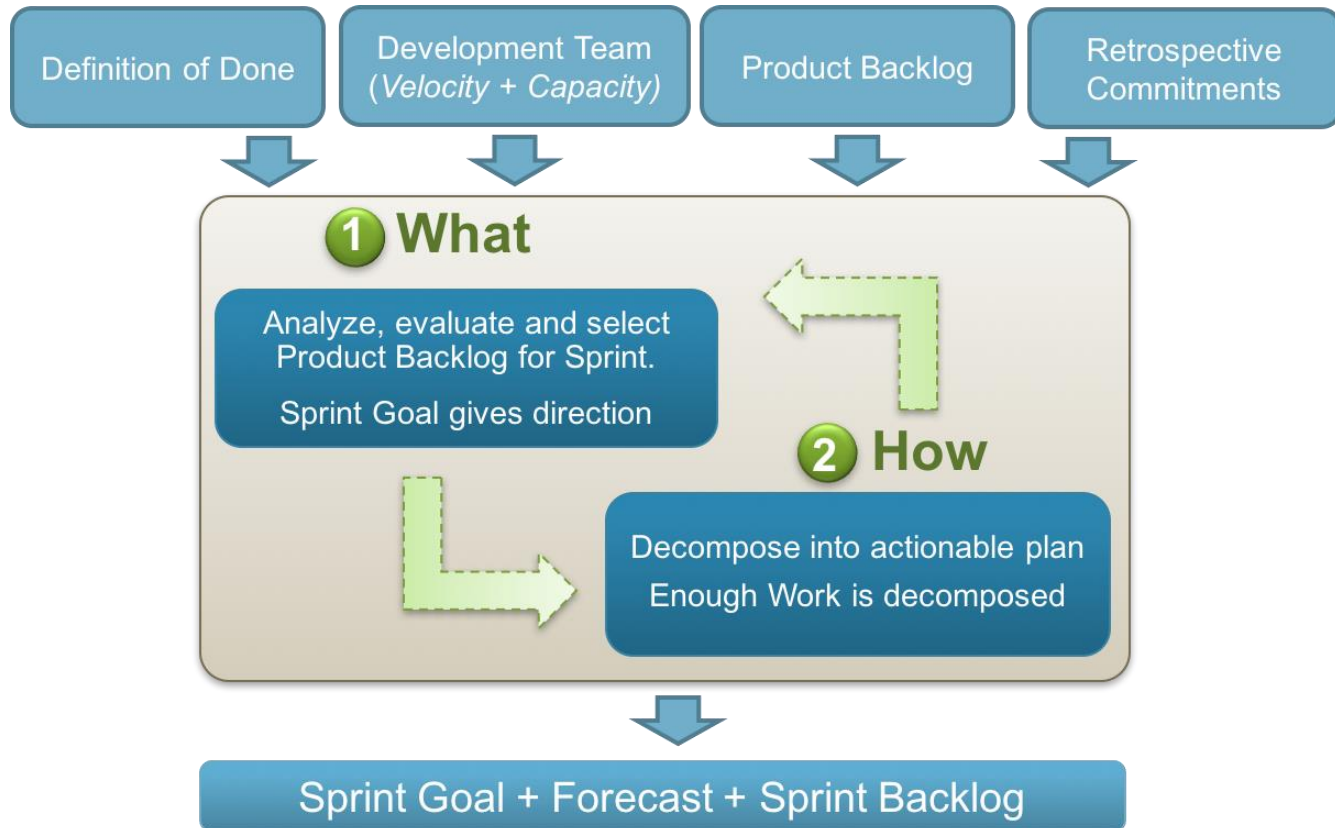


Transparency:
Giving visibility to the significant aspects of the process to those responsible for the outcome.

Inspection:
Timely checks on the progress toward a Sprint Goal to detect undesirable variances.

Adaptation:
Adjusting a process as soon as possible to minimize any further deviation or issues.

Scrum Master



“A Scrum project is only one Sprint long. A release of software may be the sum of multiple increments (and previously developed software, if any), or there may be multiple releases of software within a Sprint.

A Scrum project cannot fail, only deliver unacceptable return on investment.”

- Ken Schwaber

Team Roles

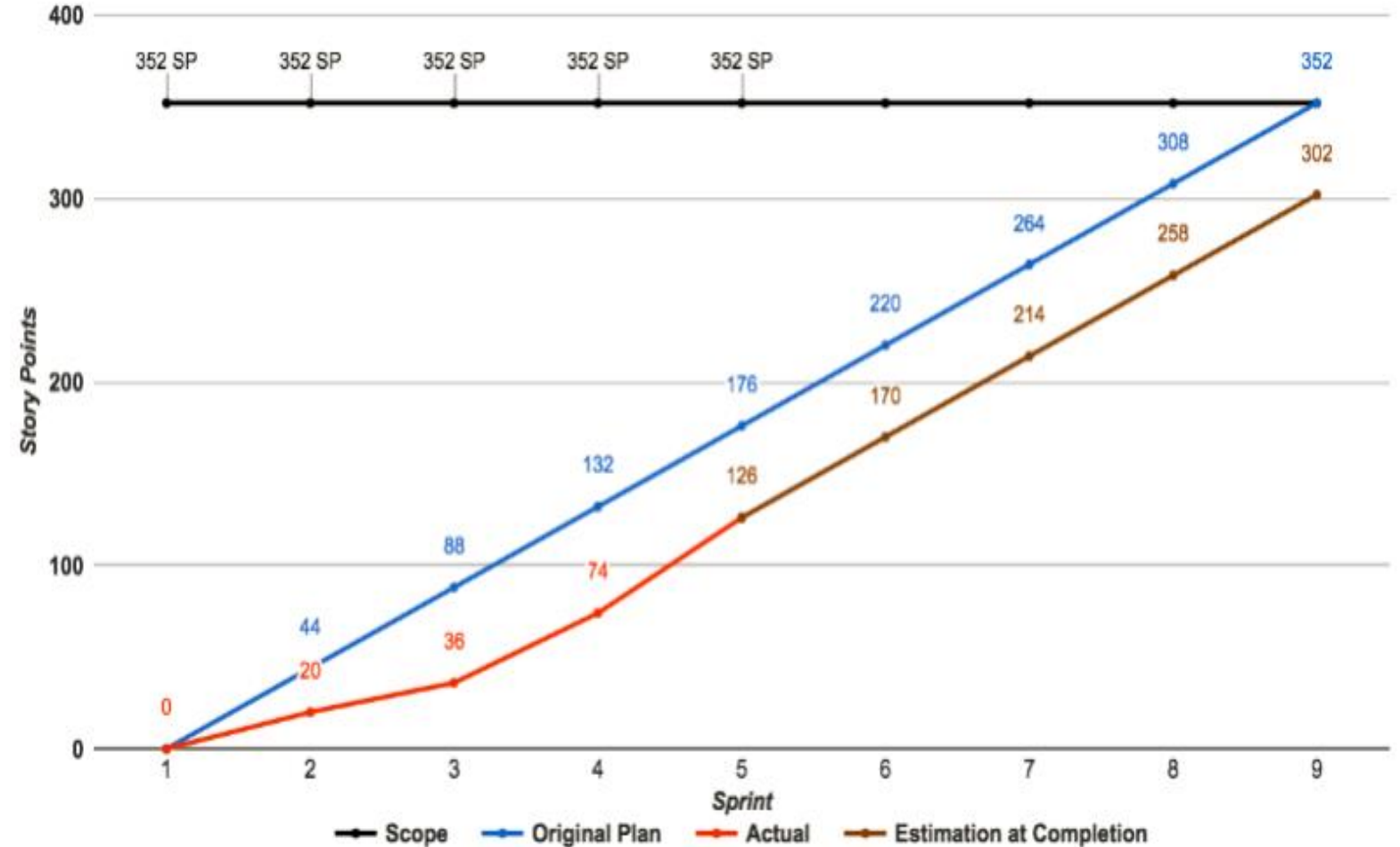
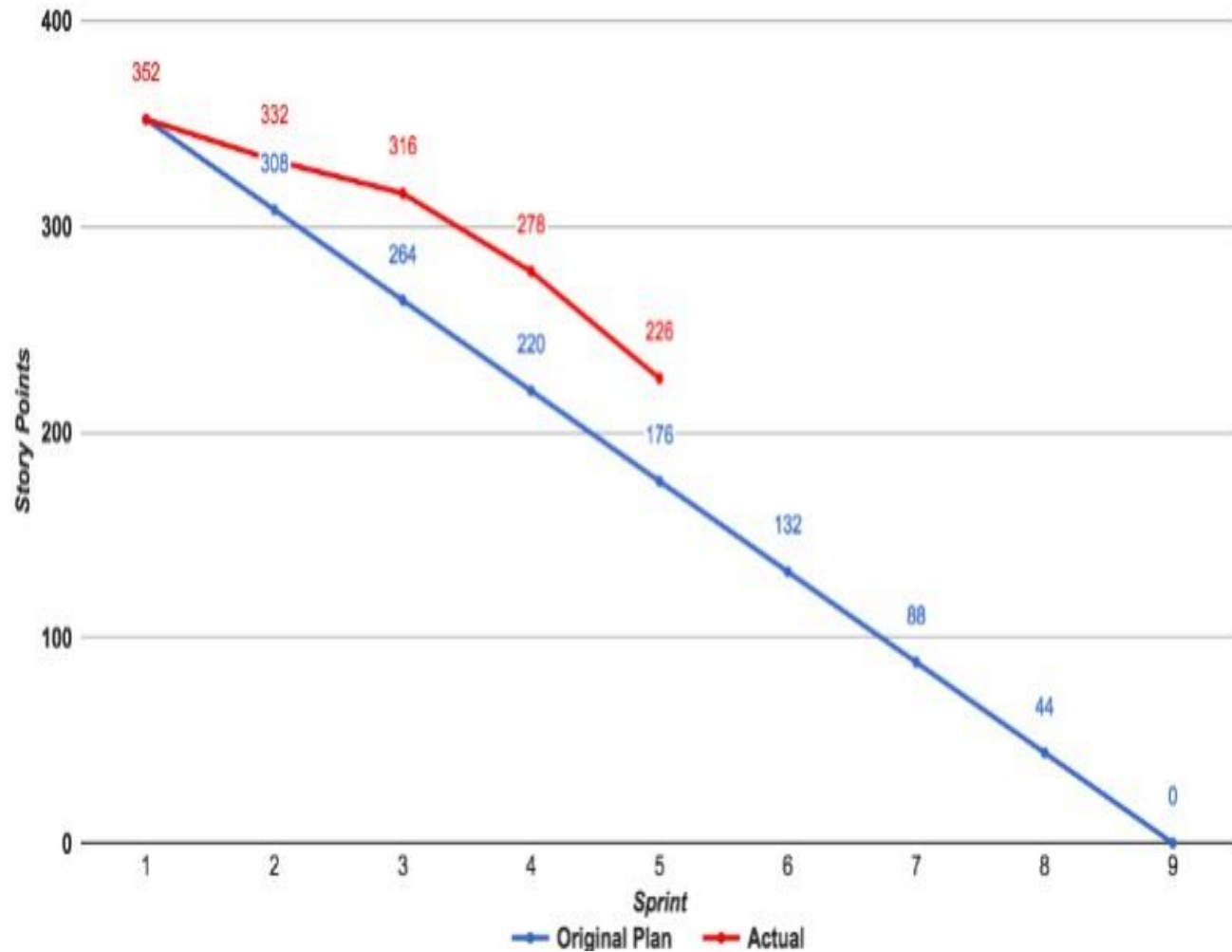


Role	Description
Cross-functional team member	<p>Cross-functional teams consist of team members with all the skills necessary to produce a working product. In software development, cross-functional teams are typically comprised of designers, developers, testers, and any other required roles. The cross-functional development teams consist of professionals who deliver potentially releasable product on a regular cadence. Cross-functional teams are critical because they can deliver finished work in the shortest possible time, with higher quality, without external dependencies.</p>
Product owner	<p>The product owner is responsible for guiding the direction of the product. Product owners rank the work based on its business value. Product owners work with their teams daily by providing product feedback and setting direction on the next piece of functionality to be developed/delivered. That means the work is small, often small enough to be described on one index card.</p> <p>The product owner works with stakeholders, customers, and the teams to define the product direction. Typically, product owners have a business background and bring deep subject matter expertise to the decisions. Sometimes, the product owner requests help from people with deep domain expertise, such as architects, or deep customer expertise, such as product managers. Product owners need training on how to organize and manage the flow of work through the team.</p> <p>In agile, the product owners create the backlog for and with the team. The backlog helps the teams see how to deliver the highest value without creating waste.</p> <p>A critical success factor for agile teams is strong product ownership. Without attention to the highest value for the customer, the agile team may create features that are not appreciated, or otherwise insufficiently valuable, therefore wasting effort.</p>
Team facilitator	<p>The third role typically seen on agile teams is of a team facilitator, a servant leader. This role may be called a project manager, scrum master, project team lead, team coach, or team facilitator.</p> <p>All agile teams need servant leadership on the team. People need time to build their servant leadership skills of facilitation, coaching, and impediment removal.</p> <p>Initially, many organizations invite external agile coaches to help them when their internal coaching capability is not yet fully developed.</p> <p>External coaches have the advantage of experience, but the disadvantage of weak relationships in the client organization. Internal coaches, on the other hand, have strong relationships in their organization, but may lack the breadth of experience that would make them highly effective.</p>

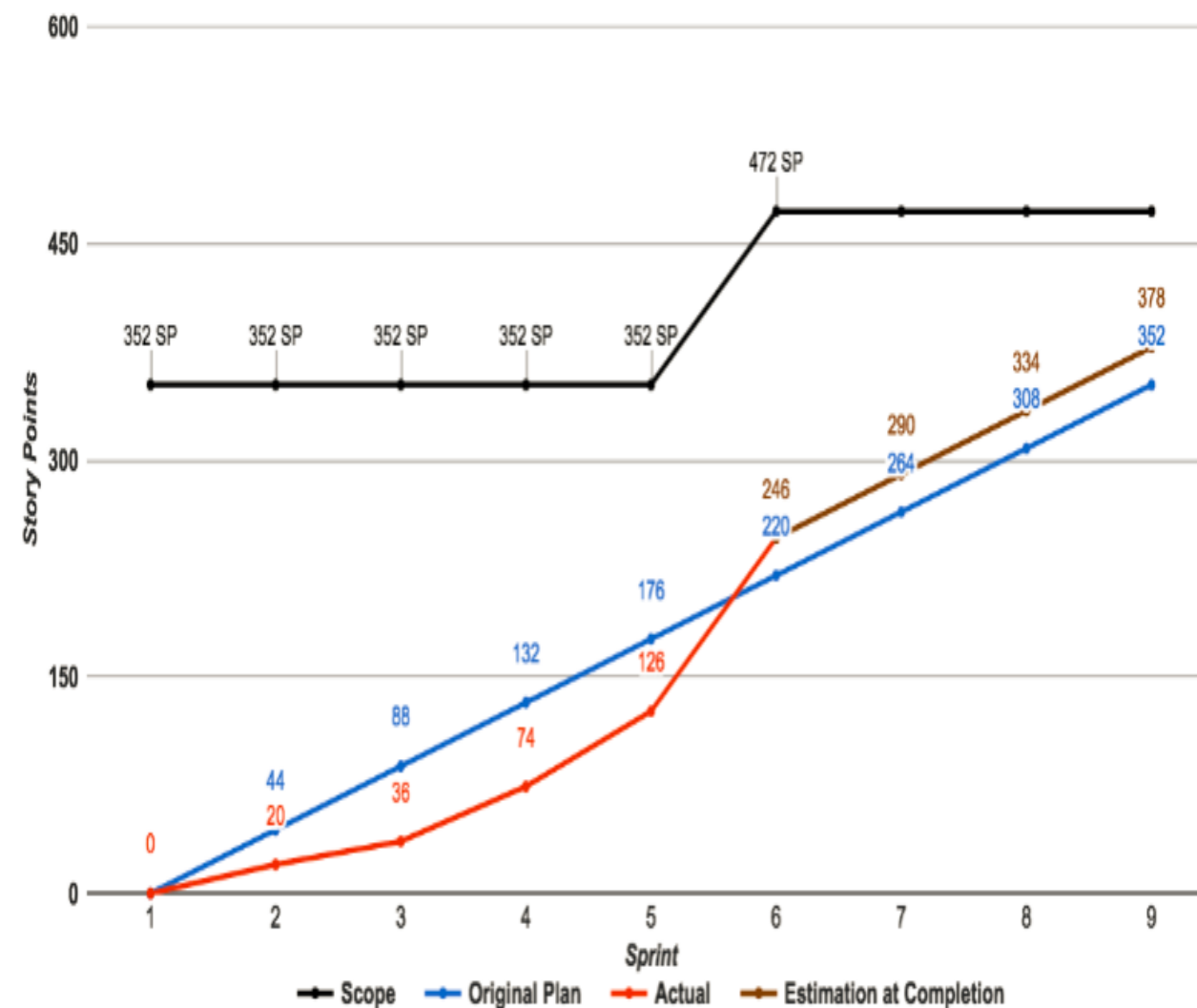
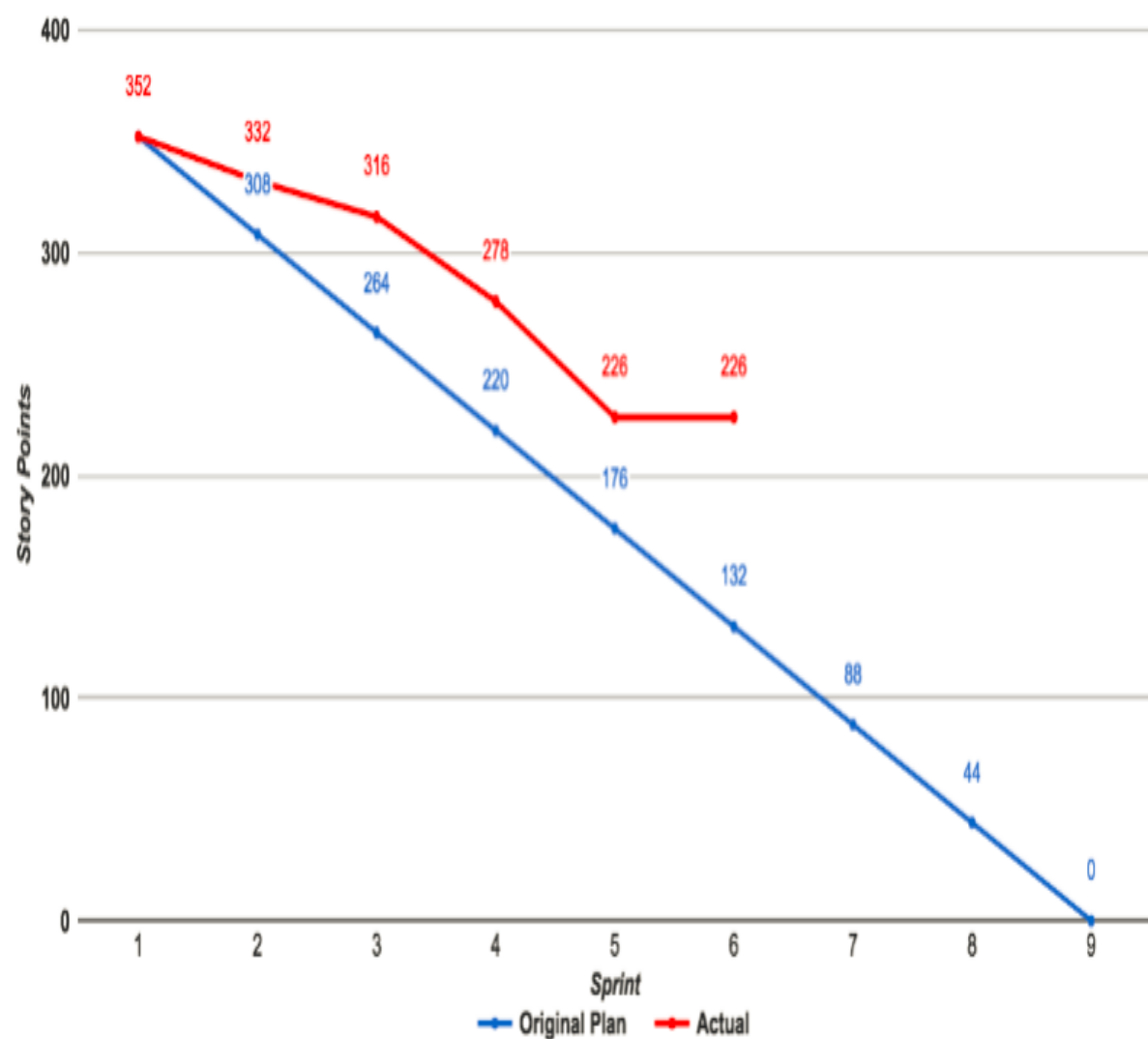


Events	Artifacts
Sprint Sprint planning Daily scrum Sprint review Sprint retrospective	Product backlog Sprint backlog Increments

Burn down Vs Burn up Charts

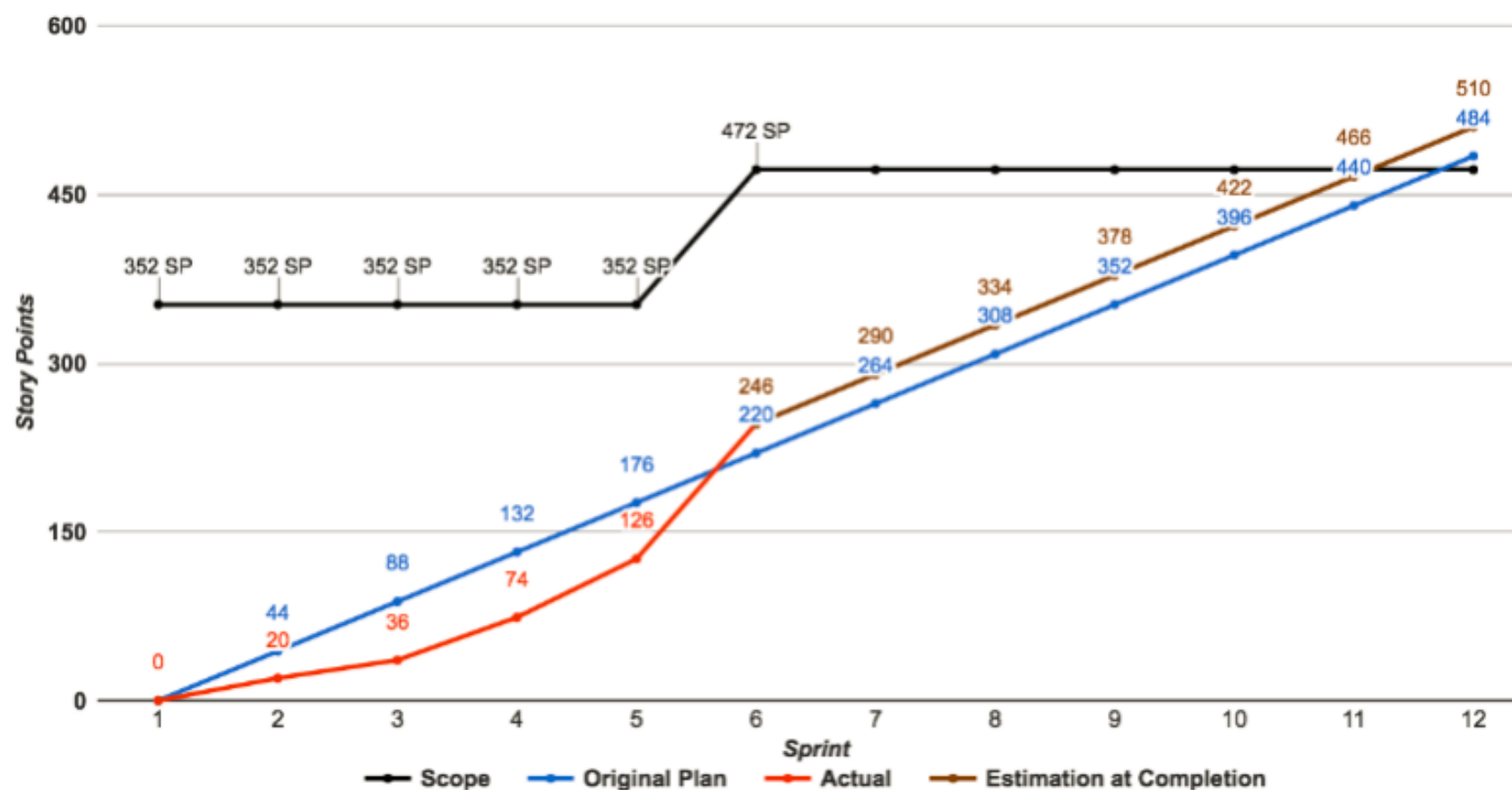


<https://stayrelevant.globant.com/en/why-you-should-use-burn-up-chart-in-agile-instead/>

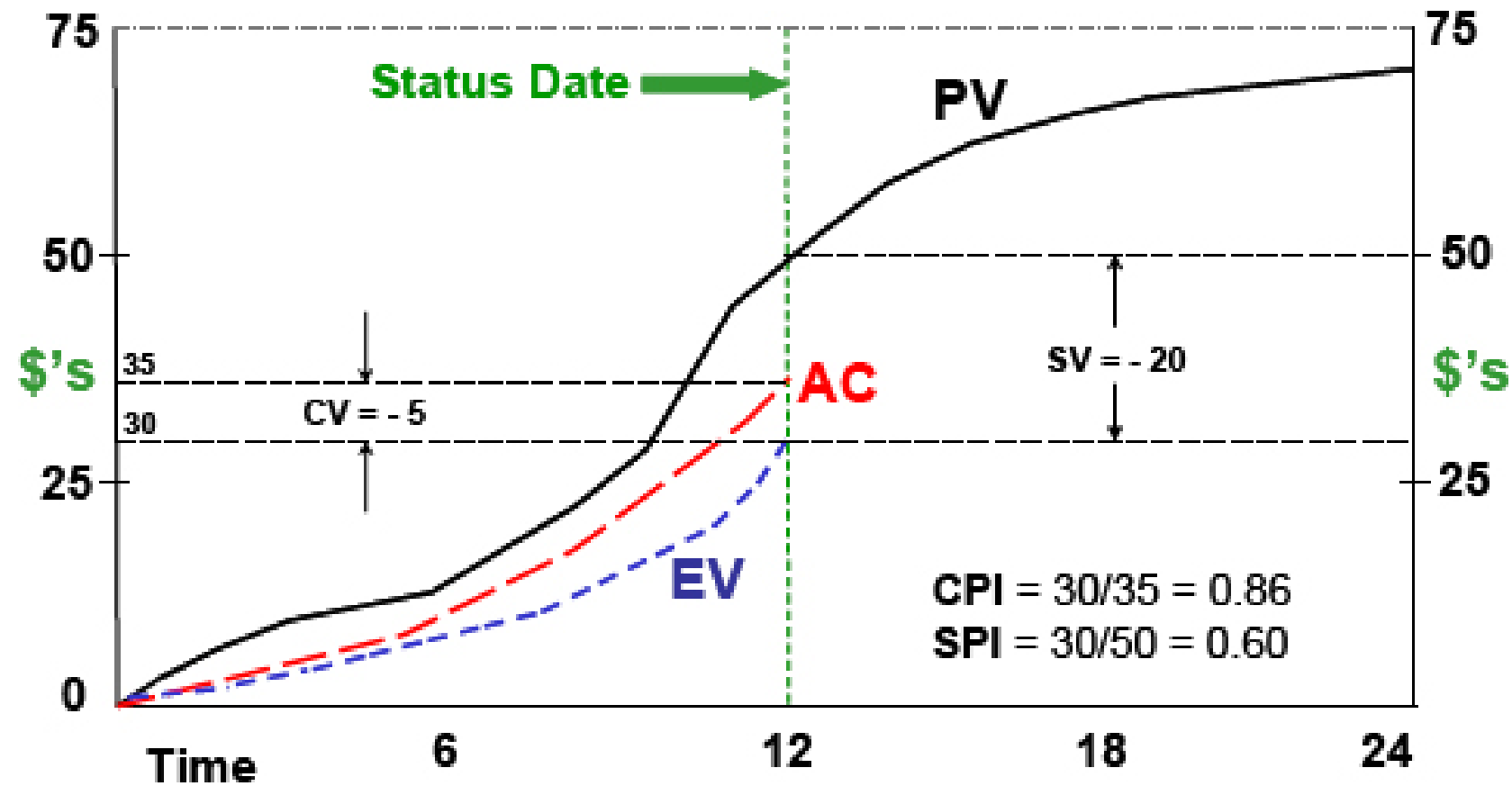


Concepto	Value
Original Scope	352
Actual Scope	472
Scope Change %	34.09%
Original Velocity (until Actual Sprint)	220
Actual Velocity (until Actual Sprint)	246
Deviation Velocity %	11.82%
EAC (Estimate at Completion)	378
Deviation EAC (Estimate at Completion)	94
Deviation EAC %	19.92%

Burn down chart	Burn Up chart
Track how much work is <u>left</u> .	Track how much work was <u>Done</u> .
Burn down is more detailed; it is a low level view, used better for day to day tracking activities.	Usually shows Work complete comparing to. Content scope/ Goals changes. It can detect changes in the scope.
	Burn up chart are better for the larger perspective. Measuring points of progress every end of sprint/period of time.
	It is a good use when our scope is a time frame and the amount of work is not necessary defined or changed.



Earned Value Model



CPI > 1	CPI = 1	CPI < 1
Under Budget	On Budget	Over Budget
EV > AC	EV = AC	EV < AC

SPI > 1	SPI = 1	SPI < 1
Ahead of Schedule	On Schedule	Behind Schedule
EV > PV	EV = PV	EV < PV

Project budget of \$ 175,000, and having completed one out of four Iterations, we have this product backlog and these actuals

Agile EVM



Feature	Estimate (storypoints)	Completed (storypoints)	Actual Cost (1000s dollars)
Welcome Screen	10	10	15
Advert - Splash Screen	20	20	30
Login Screen	10	10	20
Personalized Google Ads	20		
Catalog Browser	20		
Catalog Editor	10		
Shopping Basket Browser	5		
Shopping Card Editor	25		
Check-out Process	20		
Invoice Calculation	10		
Credit Card Verification	10		
PayPal Payment Handling	20		
Order Confirmation Email	20		
Totals	200	40	65

It is important to note that in AgileEVM there is no credit for partial completion.

The backlog items are either done or not done (**0 or 100%**.) In keeping with Agile terminology: a backlog item is only 'complete' and story points awarded when the customer accepts the item as 'done'.

EVM Metrics



Planned Value (PV) for a given iteration is the Expected Percent Complete multiplied by the Total Budget (25% of \$ 175,000 = **\$ 43,750**);

Actual Percent Complete equates to the total number of story points completed divided by the total number of story points planned (40/200 = 20% complete); **Earned Value (EV)** is calculated by multiplying Actual Percent Complete by the Total Budget (20% of \$ 175,000 = **\$ 35,000**).

Actual Cost (AC) is **\$65000** (Refer the table)

CPI is EV/AC; $35,000 / 65,000 = .53$.

Estimate at Complete is \$ 175,000 / .53 = \$ 330,188 we predict to be 47% over budget

SPI is EV/PV $35,000 / 43,750 = .8$ - we are 20% behind schedule

Estimated Completion can be estimated by dividing the SPI into the Planned Iterations, in our example we planned to finish the work in 4 Iterations and expect to need: $4 / .8 = 5$ Iterations.

Involvement Vs Commitment

Pig and Chicken Fable

The fable of the Chicken and the Pig is used to illustrate the differing levels of project stakeholders involved in a project. The basic fable runs:

A Pig and a Chicken are walking down the road.

The Chicken says: "Hey Pig, I was thinking we should open a restaurant!"

Pig replies: "Hm, maybe, what would we call it?"

The Chicken responds: "How about 'ham-n-eggs'?"

The Pig thinks for a moment and says: "No thanks. I'd be committed, but you'd only be involved."

Sometimes, the story is presented as a riddle;

Question: In a bacon-and-egg breakfast, what's the difference between the Chicken and the Pig?

Answer: The Chicken is involved, but the Pig is committed!



KANBAN

- A **LEAN** methodology
- **PULL** instead of **PUSH**
- **Reduce the Time between Concept → Cash**
- **Optimization of Throughput 'rather' Resources**
- **WIP is Bad and Necessary → Urge to Reduce**

Muda Mura Muri



Muri = overburdened



Mura = unevenness, fluctuation, variation



Muda = waste



No Muri, Mura, or Muda

Lean Principles

There are 7 lean principles which seems to agreed & practiced globally with most of the software development processes

- Eliminate Waste
- Create Knowledge
- Build Quality In
- Defer Commitment
- Optimize the whole
- Deliver Fast
- Respect people



Defer Commitment



Deliver Fast



Create Knowledge



Eliminate Waste



Build Quality In



Optimize the Whole



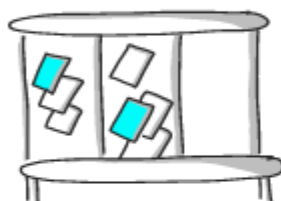
Respect People

THE SEVEN WASTES OF SOFTWARE DEVELOPMENT

©SketchingPM.com

By Matt & Max

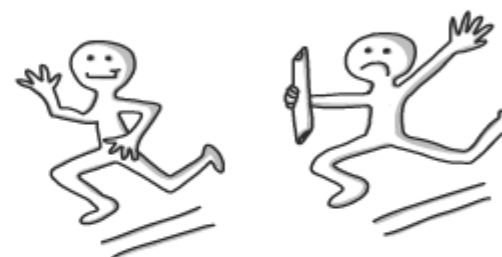
AGILE KNOWLEDGE BASE SKETCH #5



PARTIALLY DONE WORK



TASK SWITCHING



HANDOFFS



DELAYS



DEFECTS



EXTRA FEATURES



RELEARNING

Relearning referring to repeating a value adding activity spending time relearning things we have already learned.

e.g.: undocumented code force to a new developer to “relearn” what was already learned for the previews programmer; a solution of an undocumented problem solved have to be relearn when the problem reappear

Kanban: 3 simple rules

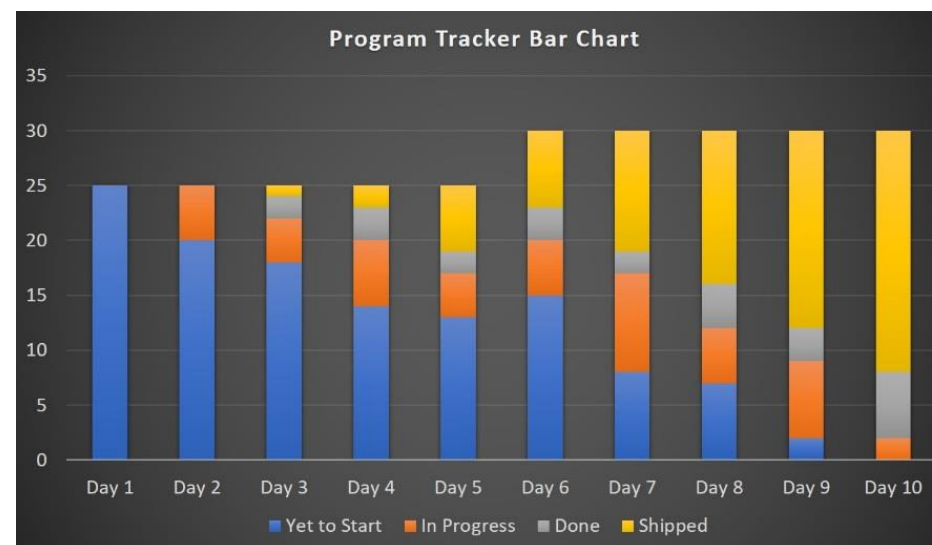
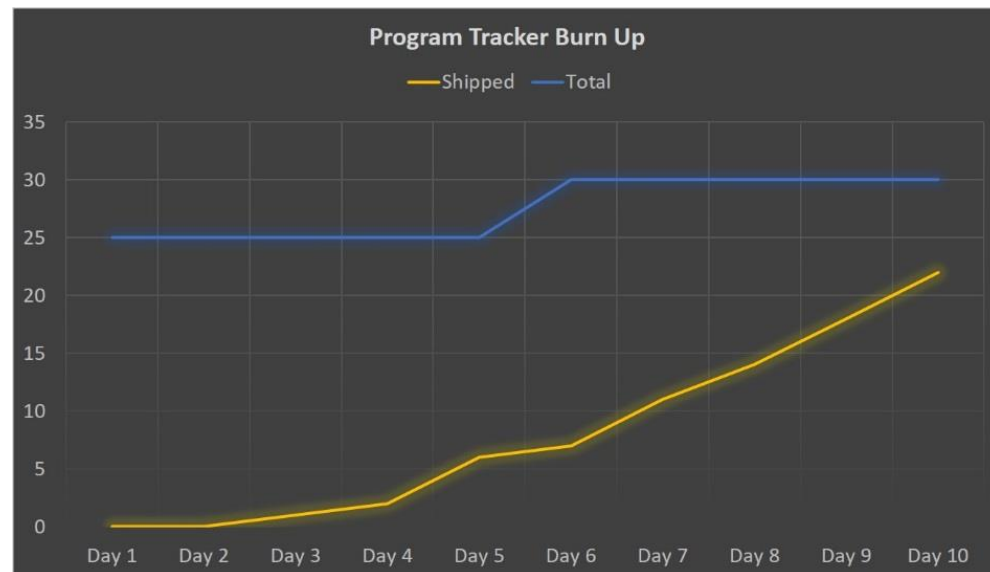
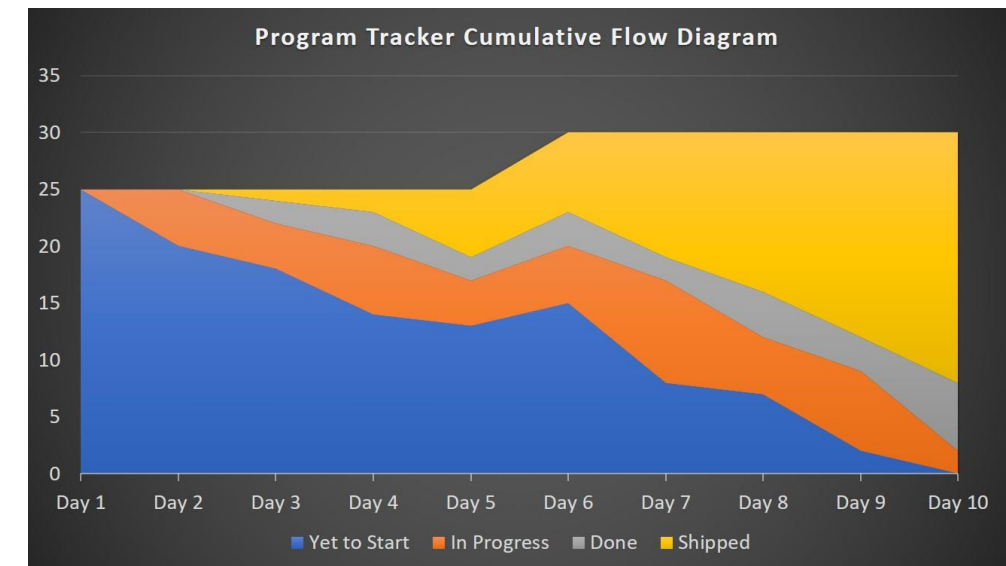
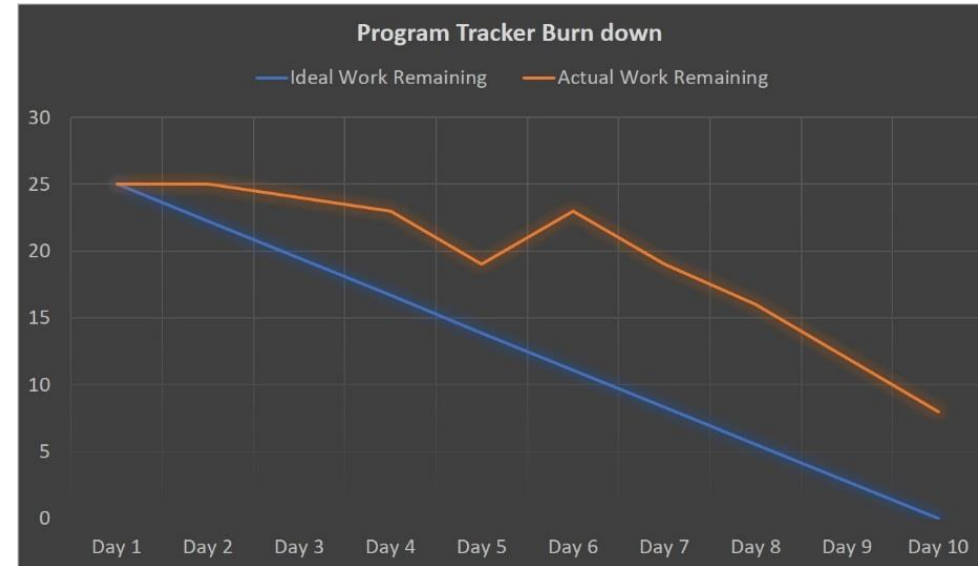


- Strict Queue limits (limit work to capacity)
- **Pull** value through (not **Push**)
- Make it Visible

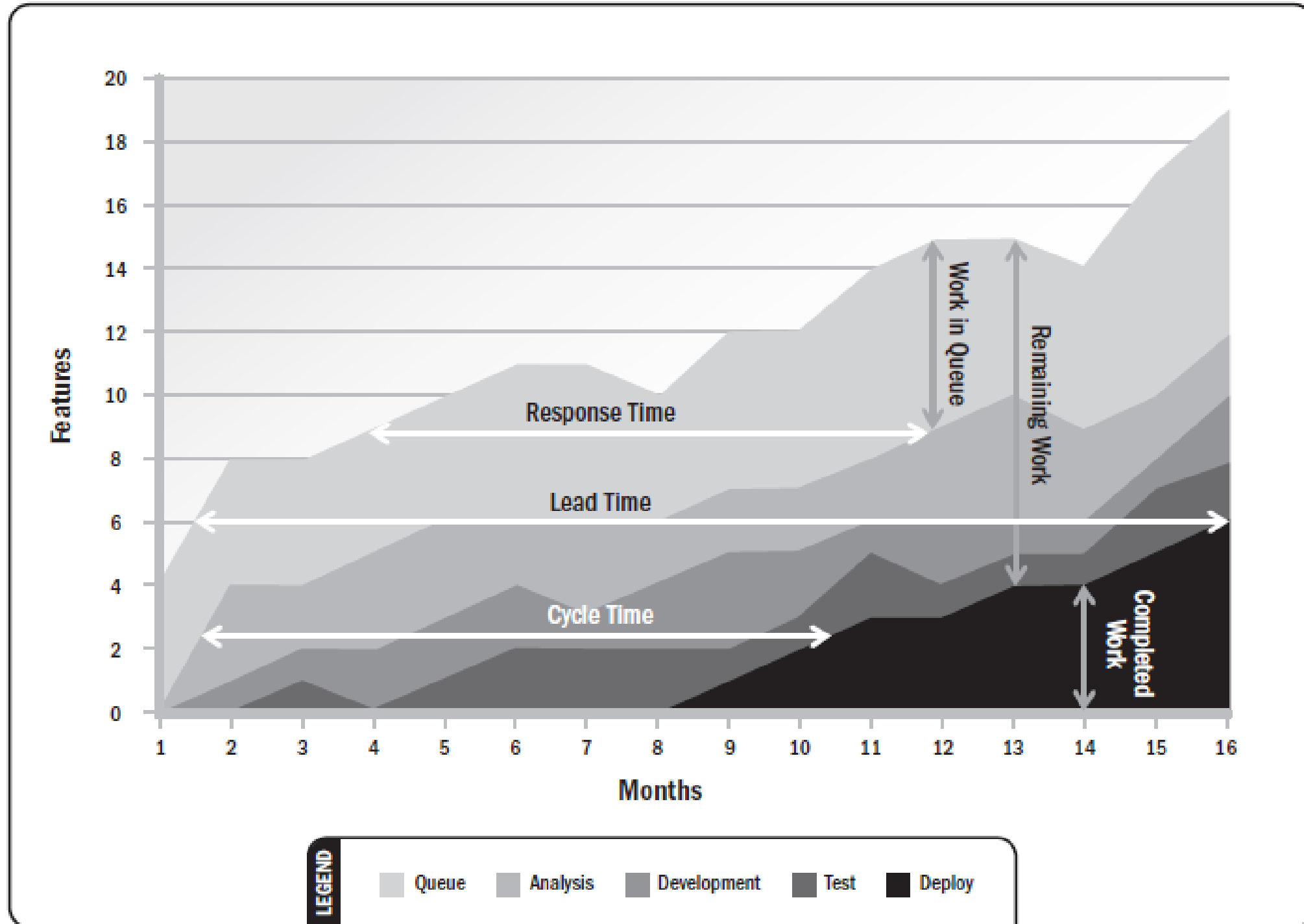
Agile Metrics – KANBAN CFD



	Yet to Start	In Progress	Done	Shipped
Day 1	25	0	0	0
Day 2	20	5	0	0
Day 3	18	4	2	1
Day 4	14	6	3	2
Day 5	13	4	2	6
Day 6	15	5	3	7
Day 7	8	9	2	11
Day 8	7	5	4	14
Day 9	2	7	3	18
Day 10	0	2	6	22

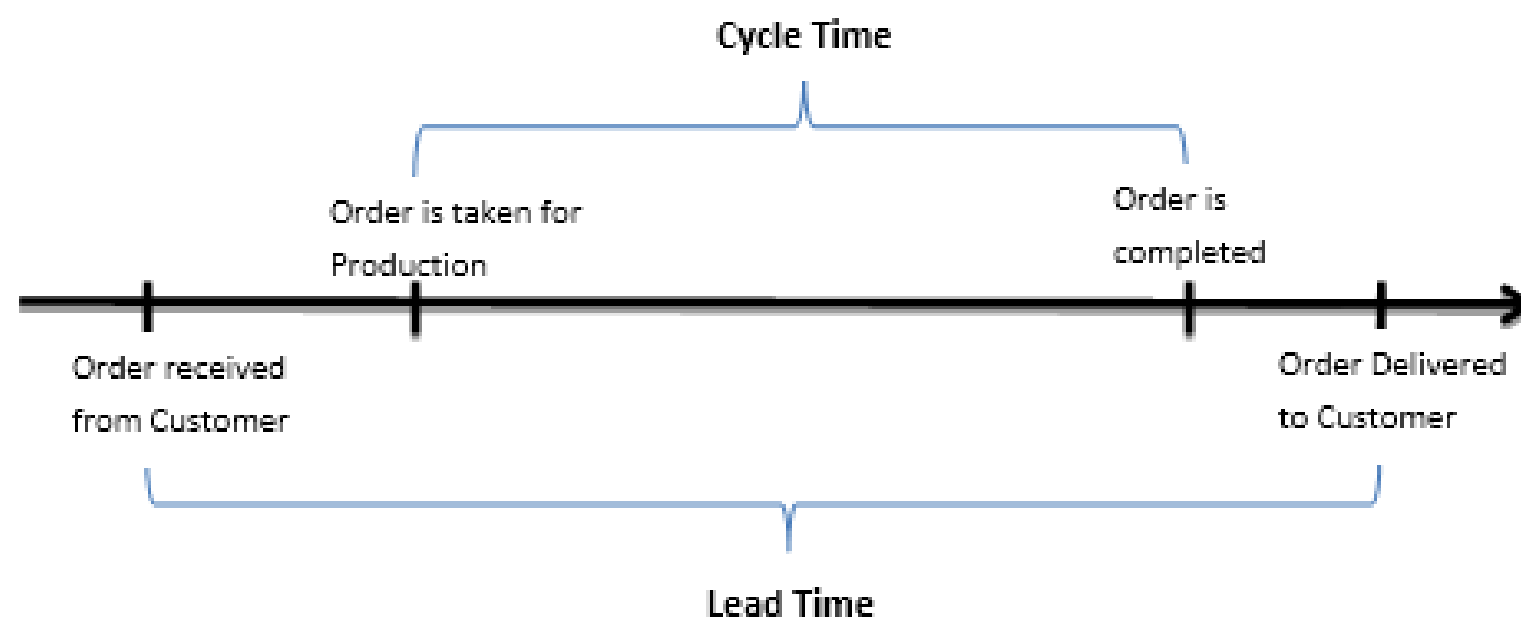


<https://www.youtube.com/watch?v=eo2uv8avEsU>



Cycle Time Vs Lead Time

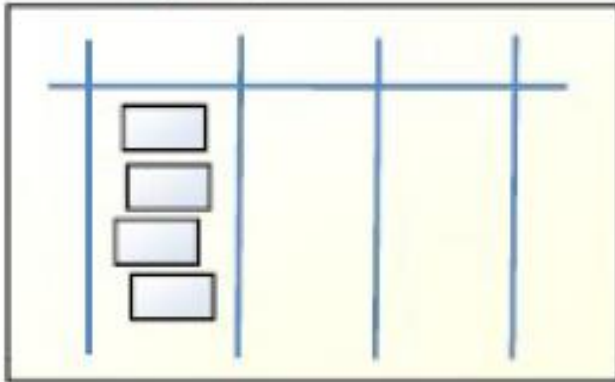
- Lead time clock starts when the request is made and ends at delivery. Cycle time clock starts when work begins on the request and ends when the item is ready for delivery. Cycle time is a more mechanical measure of process capability. Lead time is what the customer sees.
- Lead time depends on cycle time, but also depends on your willingness to keep a backlog, the customer's patience, and the customer's readiness for delivery.
- Another way to think about it is: cycle time measures the completion rate, lead time measures the arrival rate. A producer has limited strategies to influence lead time. One is pricing (managing the arrival rate), another is managing cycle time (completing work faster/slower than the arrival rate).



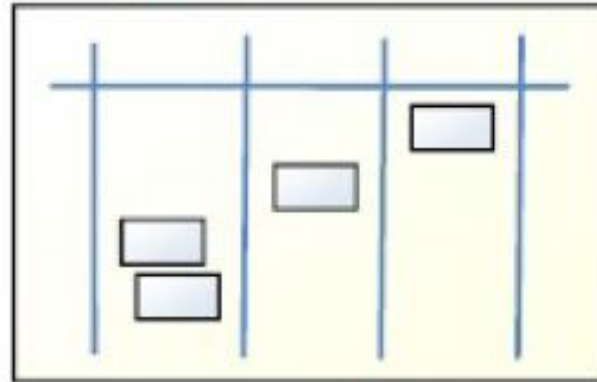
Scrum vs Kanban

Scrum

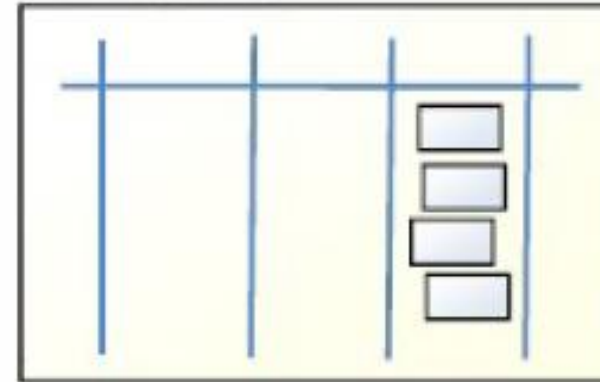
First day of sprint



Mid-sprint

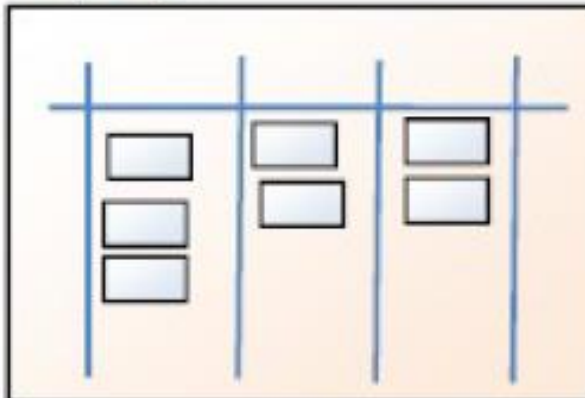


Last day of sprint



Kanban

Any day





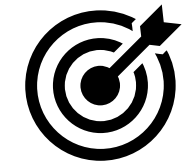
Defining Principles	Core Properties
<p>Start with current state</p> <p>Agree to pursue incremental, evolutionary change</p> <p>Respect the current process, roles, responsibilities, and titles</p> <p>Encourage acts of leadership at all levels</p>	<p>Visualize the workflow</p> <p>Limit work in progress</p> <p>Manage flow</p> <p>Make process policies explicit</p> <p>Implement feedback loops</p> <p>Improve collaboratively</p>

XP

- Simplicity
- Feedback
- Courage
- Respect

XP - Values

XP Practice Area	Primary	Secondary
Organizational	<ul style="list-style-type: none"> • Sit together • Whole team • Informative workspace 	<ul style="list-style-type: none"> • Real customer involvement • Team continuity • Sustainable pace
Technical	<ul style="list-style-type: none"> • Pair programming • Test-first programming • Incremental design 	<ul style="list-style-type: none"> • Shared code/collective ownership • Documentation from code and tests • Refactoring
Planning	<ul style="list-style-type: none"> • User stories • Weekly cycle • Quarterly cycle • Slack 	<ul style="list-style-type: none"> • Root cause analysis • Shrinking teams • Pay per use • Negotiated scope contract • Daily standups
Integration	<ul style="list-style-type: none"> • 10-minute build • Continuous integration • Test-first 	<ul style="list-style-type: none"> • Single code base • Incremental deployment • Daily deployment



XP – Ken Beck



Values

- Communication:** Everyone on a team works jointly at every stage of the project.
- **Simplicity:** Developers strive to write simple code bringing more value to a product, as it saves time and efforts.
 - **Feedback:** Team members deliver software frequently, get feedback about it, and improve a product according to the new requirements.
 - **Respect:** Every person assigned to a project contributes to a common goal.
 - **Courage:** Programmers objectively evaluate their own results without making excuses and are always ready to respond to changes.

Principles

- Rapid feedback:** Team members understand the given feedback and react to it right away.
- **Assumed simplicity:** Developers need to focus on the job that is important at the moment and follow YAGNI (You Ain't Gonna Need It) and DRY (Don't Repeat Yourself) principles.
 - **Incremental changes:** Small changes made to a product step by step work better than big ones made at once.
 - **Embracing change:** If a client thinks a product needs to be changed, programmers should support this decision and plan how to implement new requirements.
 - **Quality work:** A team that works well, makes a valuable product and feels proud of it.

EXTREME PROGRAMMING PRACTICES	
Group	Practices
Feedback	<ul style="list-style-type: none">✓ Test-Driven Development✓ The Planning Game✓ On-site Customer✓ Pair Programming
Continual Process	<ul style="list-style-type: none">✓ Continuous Integration✓ Code Refactoring✓ Small Releases
Code understanding	<ul style="list-style-type: none">✓ Simple Design✓ Collective Code Ownership✓ System Metaphor✓ Coding Standards
Work conditions	<ul style="list-style-type: none">✓ 40-Hour Week




YAGNI: You Ain't Gonna Need It

YAGNI

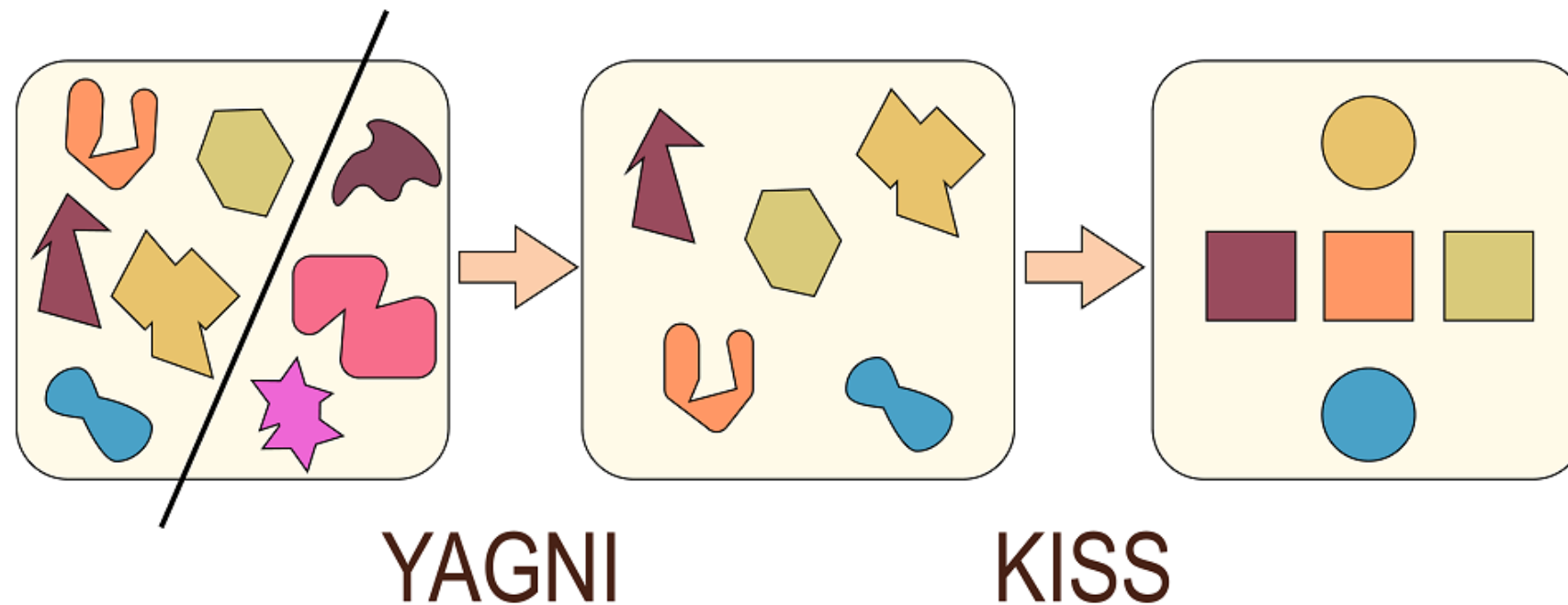
»It's currently not necessary, and we even have to maintain it!«

Code needs to be maintained. The more you have, the more complexity there will be. Adding features and capabilities that are not used (yet), wastes time twice: When you write the code and when you change or just read it. This becomes even more painful when you finally try to remove this dead code. So avoid runtime-configuration, premature optimization, and features that are only there "for the sake of completeness". If they are needed, add them later.

↑CF, ↓PSPG, ↓TP, ↓FP



design-types.net



Crystal Methodology



- Crystal is a family of methodologies for a flexible and lightweight approach to software development. The family of methodologies is color coded to differentiate its members (e.g., clear, yellow, orange, red.) The color chosen depends on the level of effort required. On one end of the spectrum is crystal clear, which is for smaller efforts, while crystal red is for larger efforts.
- Regardless of color, the crystal framework is cyclical and has three fundamental processes:
 - **chartering, delivery cycles, and wrap-up.**
- Crystal chartering includes building the team, doing an Exploratory 360, defining standards of practice for the team, and building the project plan.
- In the delivery cycle, the crystal team iteratively develops, integrates, tests, and releases the product in iterations that last from one week to two months. Like other agile frameworks, crystal includes collaborative events, like stand-up meetings and reflective improvement workshops.
- In wrap-up the team concludes the project and holds a completion ritual where the team reflects on the entire project.

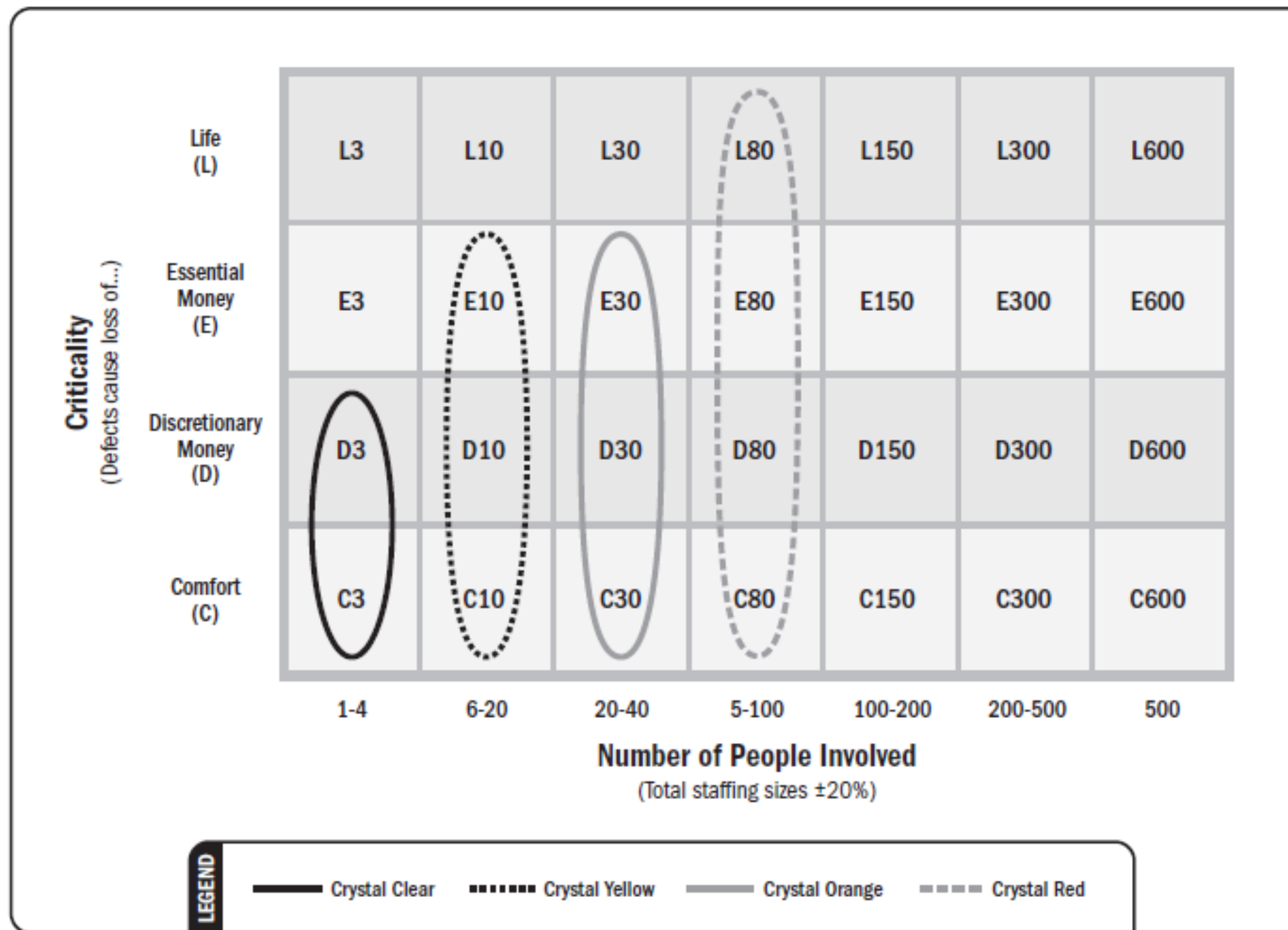


Figure A3-3. The Crystal Family of Methods

Table A3-4. The Core Values and Common Properties of Crystal

Core Values	Common Properties ^A
People Interaction Community Skills Talents Communications	Frequent delivery Reflective improvement Close or osmotic communication Personal safety Focus Easy access to expert users Technical environment with automated tests, configuration management, and frequent integration

^AThe more these properties are in a project, the more likely it is to succeed.

METRICS

AGILE KNOWLEDGE BASE SKETCH #6

TEAM'S INTUITION SAYS THAT THE LAST SPRINTS HAVE BEEN PLANNED MORE OR LESS ACCURATE

Vs.

OUR PREDICTABILITY FROM THE LAST THREE SPRINTS IS 68%

WHY METRICS? • PREDICTABILITY AND ABILITY TO PLAN

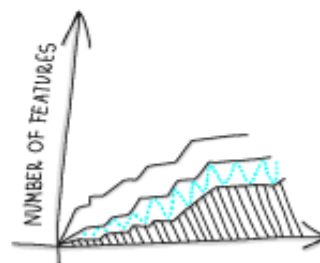


EXAMPLES

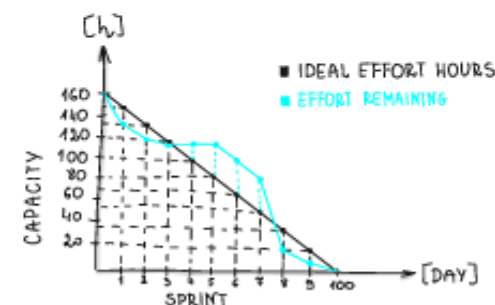
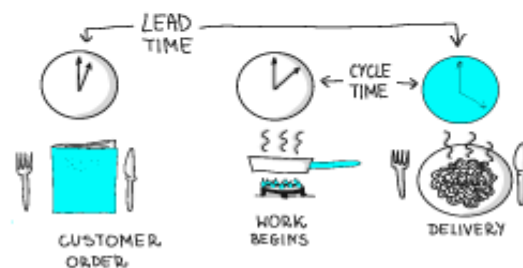
- VELOCITY
- PREDICTABILITY
- LEAD TIME - TIME FROM THE MOMENT OF ORDERING TILL FINISHING
- CYCLE TIME - DURANCE OF WORK
- CODE COVERAGE
- NUMBER OF REPORTED BACKLOG ERRORS
- HAPPINESS INDEX
- TECHNICAL METRICS VS. BUSINESS METRICS



START WITH ONE METRIC



CUMULATIVE FLOW DIAGRAM



BURN-DOWN CHART

JULY 2018							
	1	2	3	4	5	6	7
ERIC	☺	☹	🚧	🚧	☺	☹	🟢
MEIKE	☺	☹	🚧	🚧	🟢	☹	☺
MARIO	☹	☹	🚧	🚧	☹	☹	🟢
ADAM	🟢	☹	🚧	🚧	☹	☹	🟢
LOUIS	☹	☹	🚧	🚧	☹	☹	☹

HAPPINESS / MORALE INDEX

Metrics - Tools



Including but not limited to:

- velocity/throughput/productivity
- cycle time
- lead time
- EVM for agile projects
- defect rate
- approved iterations
- work in progress



OTHERS

- SCRUMBAN
- FDD
- DYNAMIC SYSTEMS DEVELOPMENT
- AUP
- ATDD

Acceptance Test Driven Development - ATDD



- The iterative cycle of ATDD with its four steps can be remembered as the four Ds: 1) Discuss, 2) Distill, 3) Develop, and 4) Demo.
- 1) Discuss: The agile team and customer or business stakeholder discuss a user story in detail. Talking about the expected behaviors the user story should have and what it should not.
- 2) The development team takes those items learned from the discussion and distills them into tests that will verify and validate those behaviors. The distillation process is where the entire team should have a good understanding of what “done” (or completed) means for a user story. What is, what the acceptance criteria are.
- 3) After distillation, the team develops the test code and product code to implement the product features.
- 4) Once the product features have been developed, the team demonstrates them to the customer or business stakeholders for feedback.

SCALED AGILE

- SCRUM OF SCRUM (SoS)
- SAFe
- LeSS
- Disciplined Agile (DA)

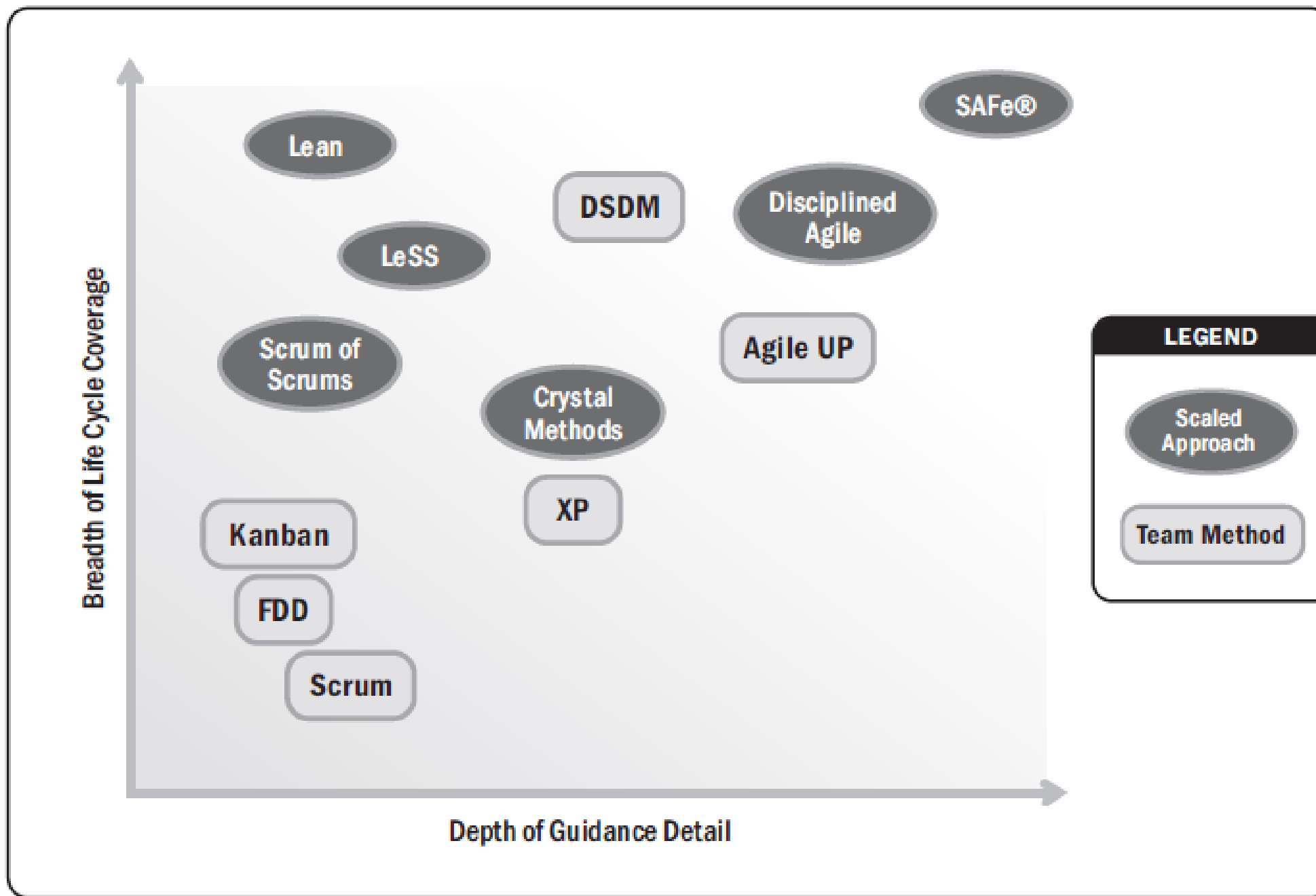


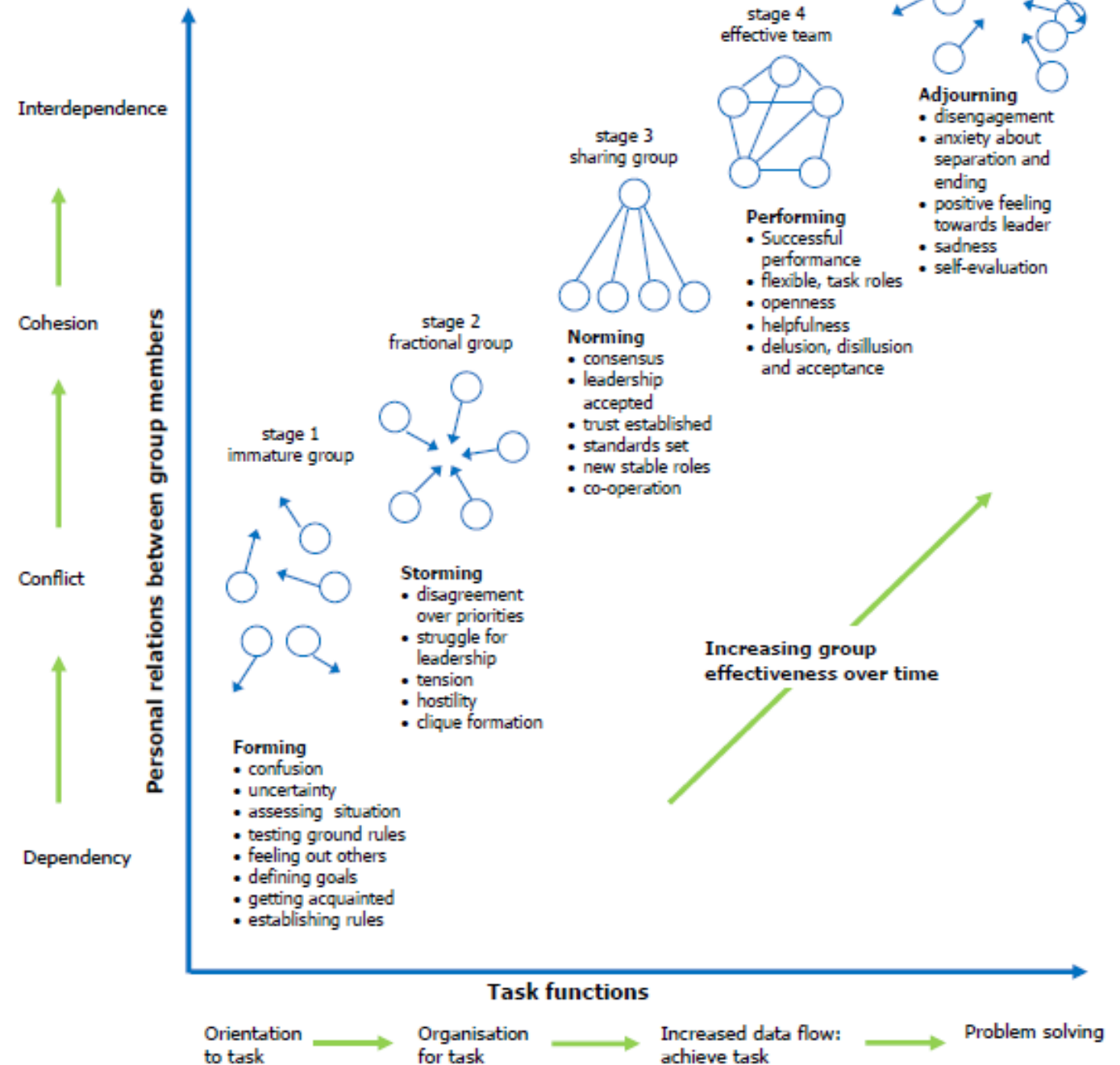
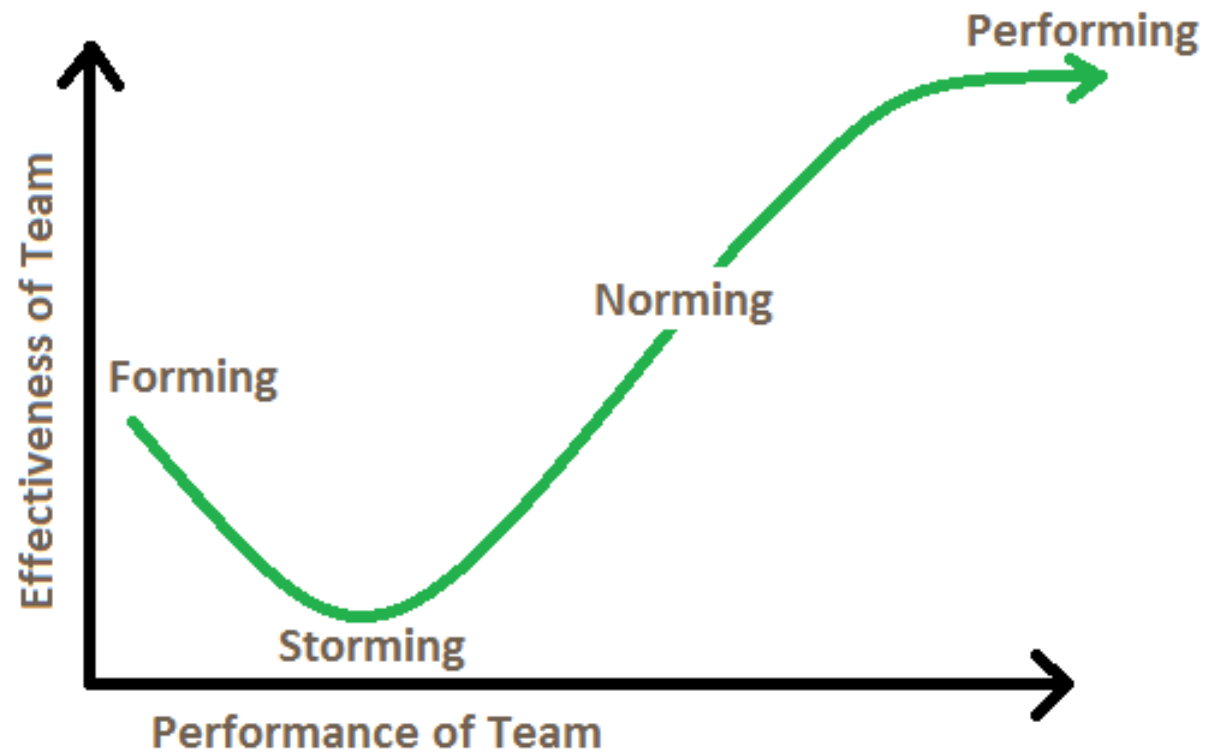
Figure A3-1. Agile Approaches Plotted by Breadth and Detail

BUILD HIGH PERFORMING TEAM

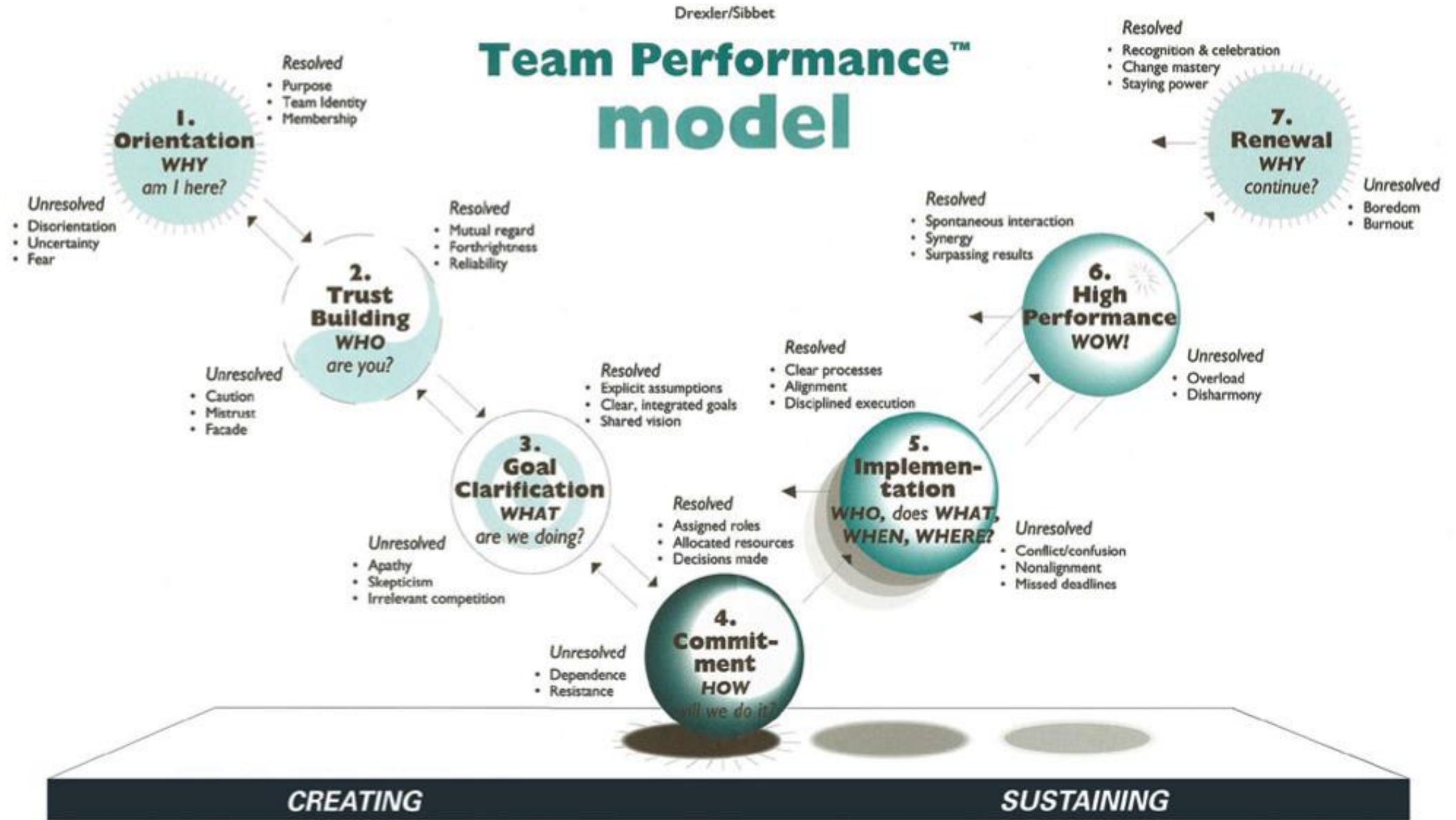
High Performing Team Bruce Tuckman's Model

Stages of group development

Tuckman's Team & Group Development Model



Drexler-Sibbet Model



1. Orientation

When teams are forming everybody wonders *WHY* they are here, what their potential fit is and whether others will accept them. People need some kind of answer to continue.

2. Trust Building

Next, people want to know *WHO* they will work with—their expectations, agendas and competencies. Sharing builds trust and a free exchange among team members.

3. Goal Clarification

The more concrete work of the team begins with clarity about team goals, basic assumptions and vision. Terms and definitions come to the fore. *WHAT* are the priorities?

4. Commitment

At some point discussions need to end and decisions must be made about *HOW* resources, time, staff—all the bottom line constraints—will be managed. Agreed roles are key.

5. Implementation

Teams turn the corner when they begin to settle on *WHO* does *WHAT*, *WHEN*, and *WHERE* in action. Timing and scheduling dominate this stage.

6. High Performance

When methods are mastered, a team can begin to change its goals and flexibly respond to the environment. The team can say, "WOW!" and surpass expectations.

7. Renewal

Teams are dynamic. People get tired; members change. People wonder " *WHY* continue?" It's time to harvest learning and prepare for a new cycle of action.

The Satir Change Model

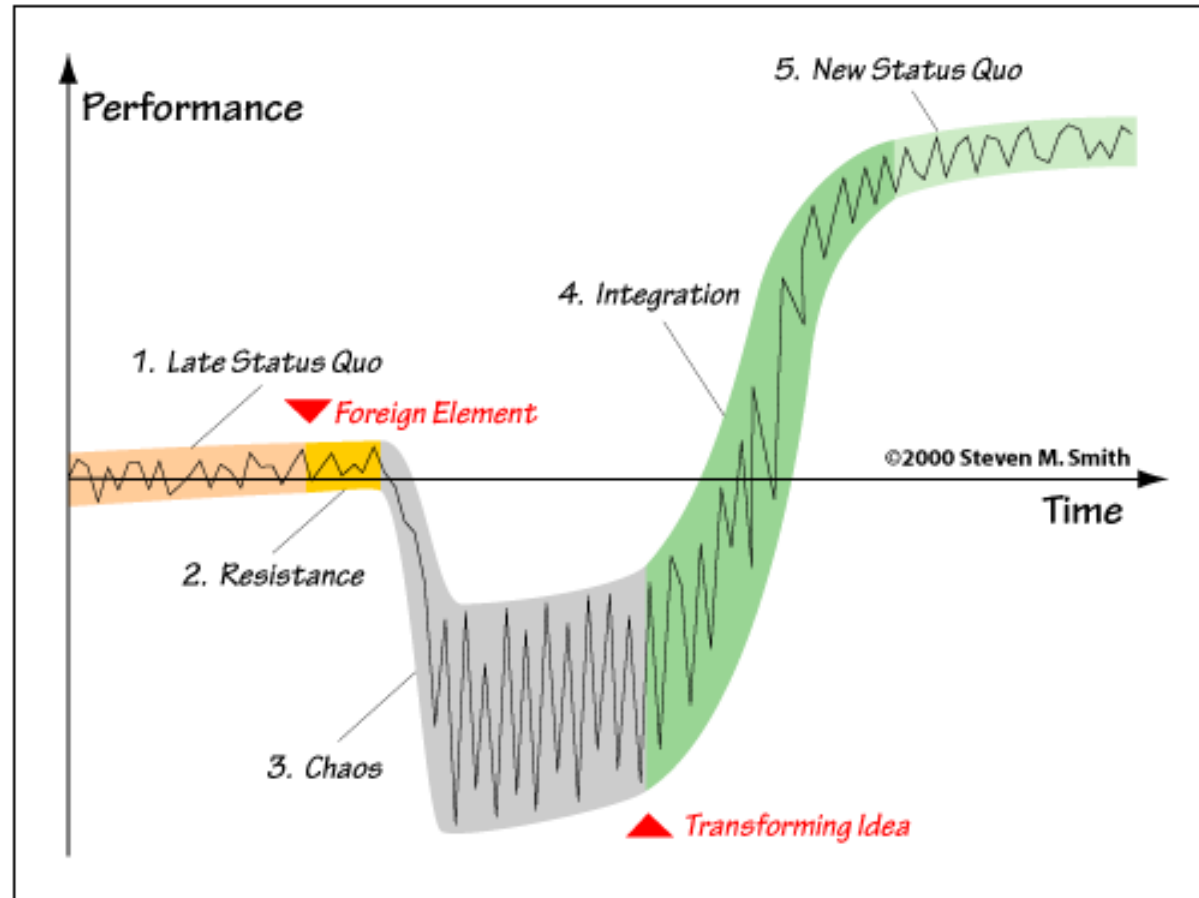


Figure 1. The impact on group performance of a well assimilated change during the five stages of the Satir Change Model.

Stage	Description	How to Help
1	Late Status Quo	Encourage people to seek improvement information and concepts from outside the group.
2	Resistance	Help people to open up, become aware, and overcome the reaction to deny, avoid or blame.
3	Chaos	Help build a safe environment that enables people to focus on their feelings, acknowledge their fear, and use their support systems. Help management avoid any attempt to short circuit this stage with magical solutions.
4	Integration	Offer reassurance and help finding new methods for coping with difficulties.
5	New Status Quo	Help people feel safe so they can practice.

Table 1. Actions for each stage that will help a group change more quickly and effectively.

Virginia Satir, a pioneer of family therapy

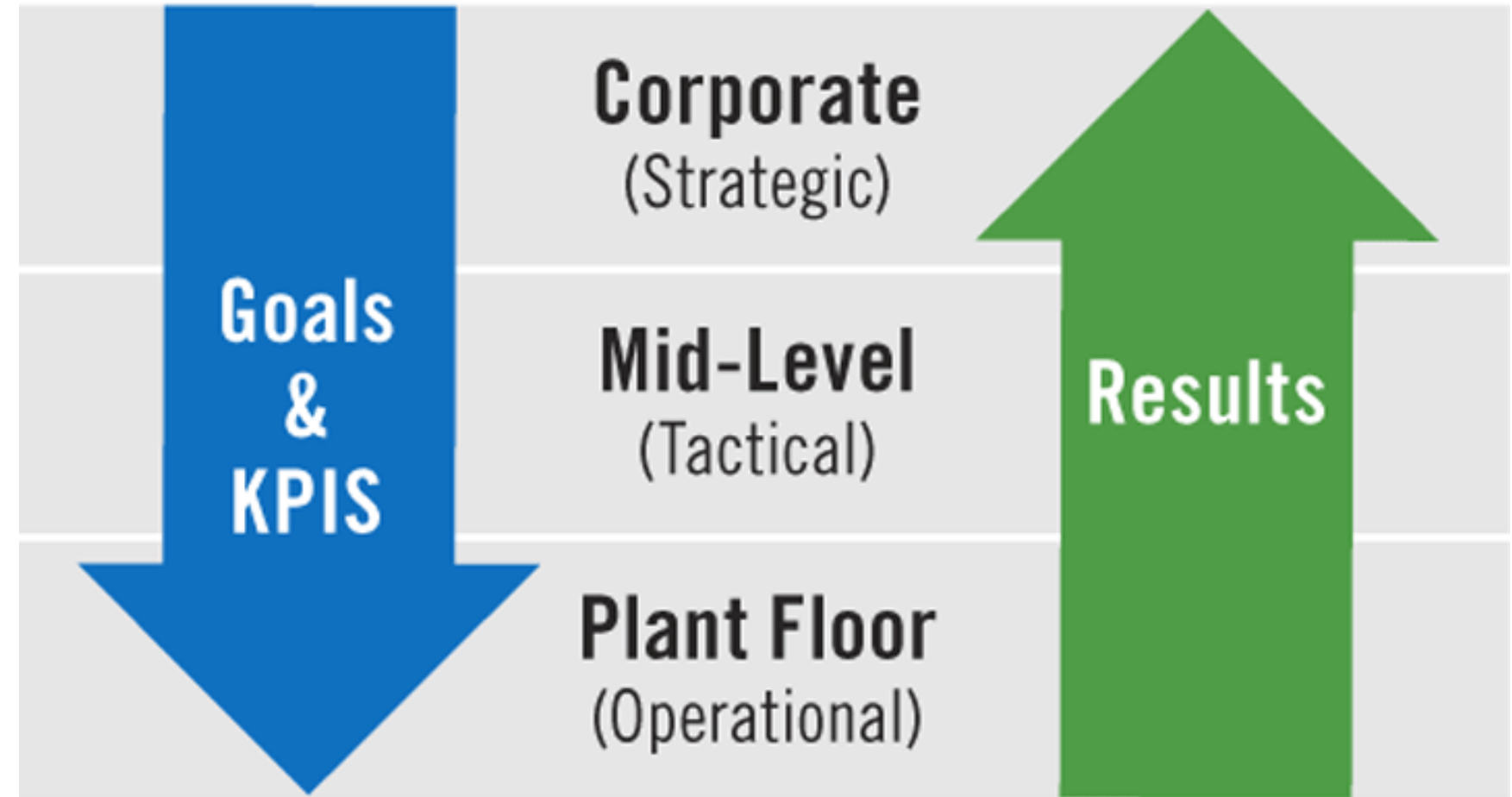
Emotional Quotient



Copyright 2012-2016 SmartTribes® Institute, LLC www.SmartTribesInstitute.com 415-320-6580

Hoshin Kanri

- Hoshin Kanri (also called Policy Deployment) is a method for ensuring that the strategic goals of a company drive progress and action at every level within that company. This eliminates the waste that comes from inconsistent direction and poor communication.
- Hoshin Kanri strives to get every employee pulling in the same direction at the same time. It achieves this by aligning the goals of the company (Strategy) with the plans of middle management (Tactics) and the work performed by all employees (Operations).



Horn Vs Halo



Common “errors”

- Halo/horn effect
 - the employee is extremely competent (or low performing) in one area and is therefore rated high (or low) in all categories
- Recency
 - the appraiser gives more weight to recent occurrences and discounts earlier performance during the appraisal period
- Bias
 - the appraiser’s values, beliefs, prejudices distort the evaluation

From The SHRM® Learning System, 2000, Module Two, General Employment Practices

Listening Levels



Co-active coaching

H. Kimsey-House, K. Kimsey-House, P. Sandahl, and L. Whitworth

- Level 1 Listening : **Internal** Listening
 - We listen to the words of the other person, but our attention is on what is means to us personally. Receiving information from yourself.
- Level 2 Listening : **Focussed** Listening
 - Sharp focus on the other person, not much awareness of the outside world.
 - Receiving information from the other one.
- Level 3 Listening : **Global** listening
 - Receiving information from everywhere, you and the other one being at the center of the universe. Greater access to your intuition. Also called environmental listening. Adjusting behaviour : performers, actors ...

Brook's Law



- Brooks' law is an observation about software project management according to which "adding human resources to a late software project makes it later".^{[1][2]} It was coined by Fred Brooks in his 1975 book The Mythical Man-Month. According to Brooks, there is an incremental person who, when added to a project, makes it take more, not less time. This is similar to the general law of diminishing returns in economics
- Exceptions and Solutions
 - Apply only to the Projects that are running late
 - Where scheduling mistakes inherent
 - CI/CD, TDD, IID practices exists

 - A way to finish a project is to invert Brooks' Law. This is the **Bermuda plan**, when 90% of the developers are removed ("send them to Bermuda") and the remaining 10% complete the software

Goodhart's Law



- This refers to the adage that any measure that becomes the target of a policy eventually stops being a useful measure. This is because people simply trying to meet the target, as gauged by a certain measure, may fail to focus on the underlying basis of the target. For instance, a government that is solely focused on boosting gross domestic product may decide to engage in wasteful spending just to boost GDP even though such spending does not improve the living standards of citizens. It is named after British economist Charles Goodhart who proposed it while serving as the chief economic adviser to the Bank of England.
- If you chase the 'Fixed Velocity Target', we would end up not focusing on the value delivery

Dreyfus Model

Dreyfus Model Of Skill Acquisition

Expert

1. Transcends reliance on rules, guidelines, and maxims
2. Intuitive grasp of situations based on deep understanding
3. Has a vision of what is possible
4. Uses an analytical approach in new situations

Proficient

1. Holistic view of situation
2. Prioritizes importance of aspects
3. Perceives deviations from the normal pattern
4. Employs maxims for guidance, with meanings that adapt to the situation at hand

Competent

1. Coping with crowdedness (multiple activities, accumulation of information)
2. Some perception of actions in relation to goals
3. Deliberate planning
4. Formulates routines

Advanced Beginner

1. Limited situational perception
2. All aspects of work treated separately with equal importance

Novice

1. Rigid adherence to taught rules or plans
2. No exercise of discretionary judgment

stevefitz.com



www.AgileVelocity.com
info@agilevelocity.com
[@agile_velocity](https://twitter.com/agile_velocity)

Death Spiral of Change

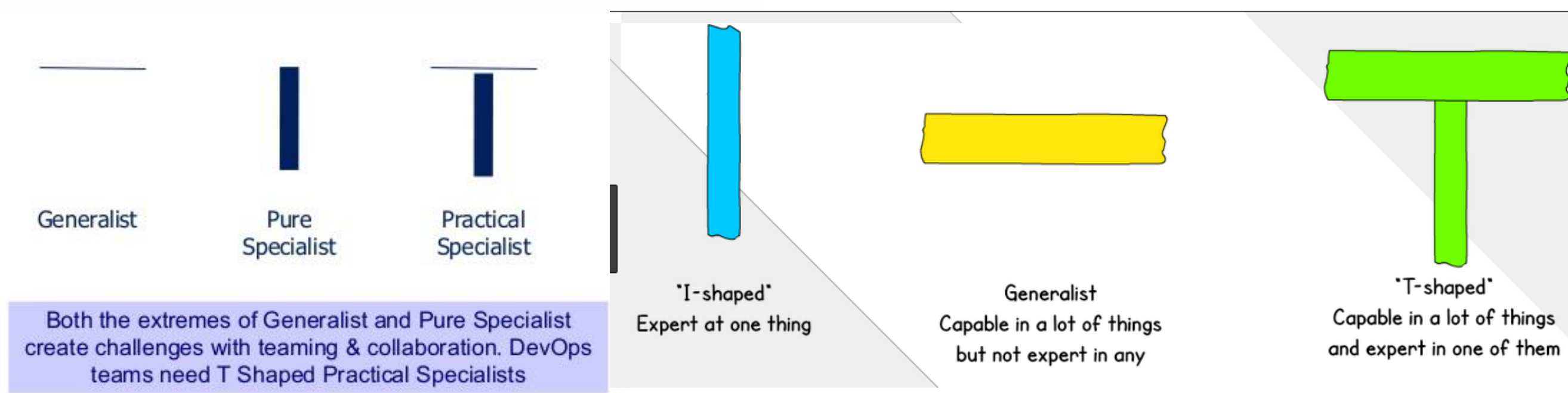


www.AgileVelocity.com
info@agilevelocity.com
[@agile_velocity](https://twitter.com/agile_velocity)

Path to Agility™ New Status Quos

Specialists Vs Generalists Vs Generalized Specialists

DevOps = T not I Shaped Specialists



Enabling Flexible & Skilled Cross-Functional Teams

©Pink Elephant, 2017. All Rights Reserved.

13



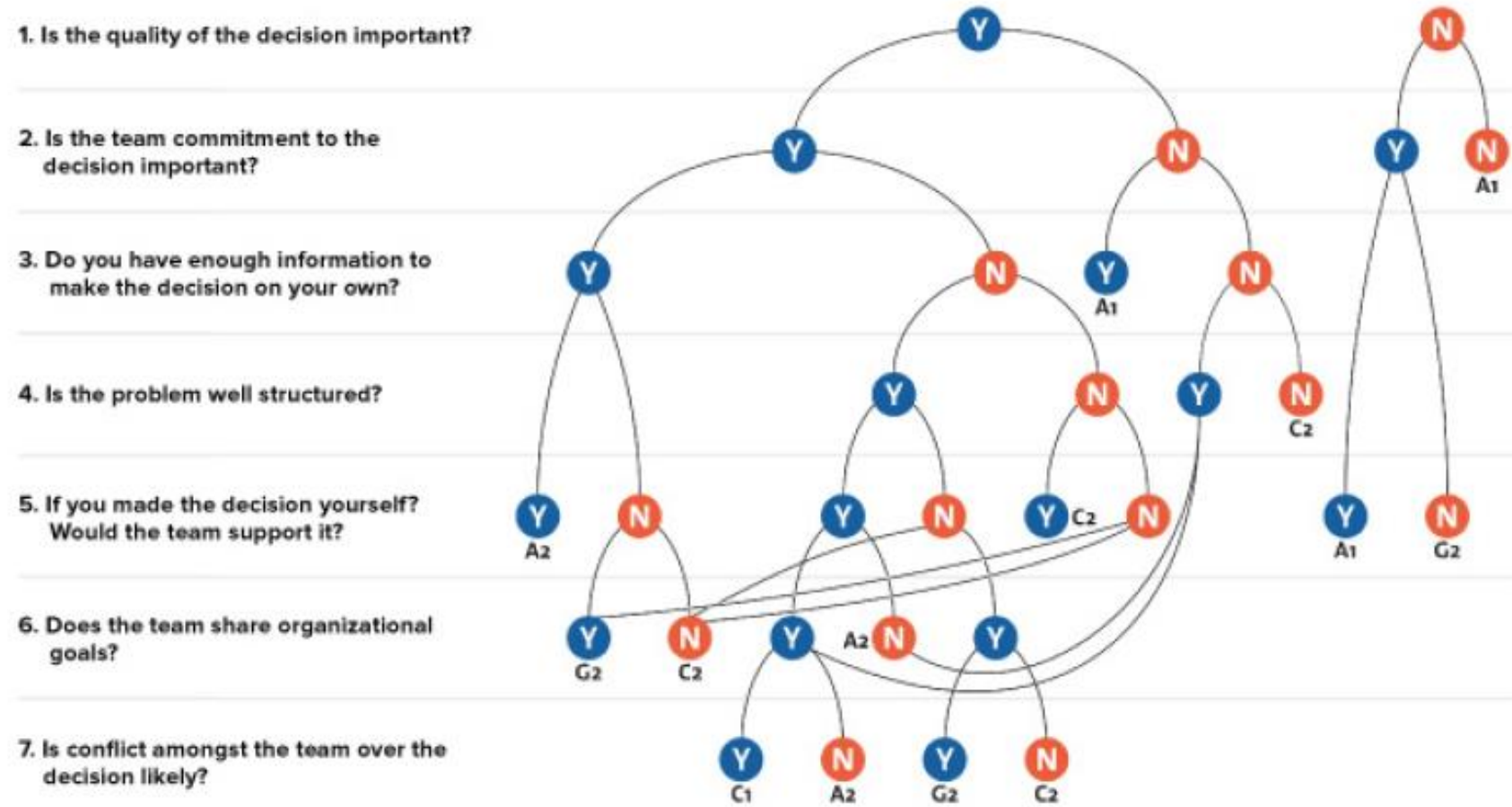
Attributes for successful Agile teams



Attribute	Goal
Dedicated people	<ul style="list-style-type: none">• Increased focus and productivity• Small team, fewer than ten people
Cross-functional team members	<ul style="list-style-type: none">• Develop and deliver often• Deliver finished value as an independent team• Integrate all the work activities to deliver finished work• Provide feedback from inside the team and from others, such as the product owner
Colocation or ability to manage any location challenges	<ul style="list-style-type: none">• Better communication• Improved team dynamics• Knowledge sharing• Reduced cost of learning• Able to commit to working with each other
Mixed team of generalists and specialists	<ul style="list-style-type: none">• Specialists provide dedicated expertise and generalists provide flexibility of who does what• Team brings their specialist capabilities and often become generalizing specialists, with a focus specialty plus breadth of experience across multiple skills
Stable work environment	<ul style="list-style-type: none">• Depend on each other to deliver• Agreed-upon approach to the work• Simplified team cost calculations (run rate)• Preservation and expansion of intellectual capital

Participatory Decision Model


The Vroom Yetton Model



KEY:

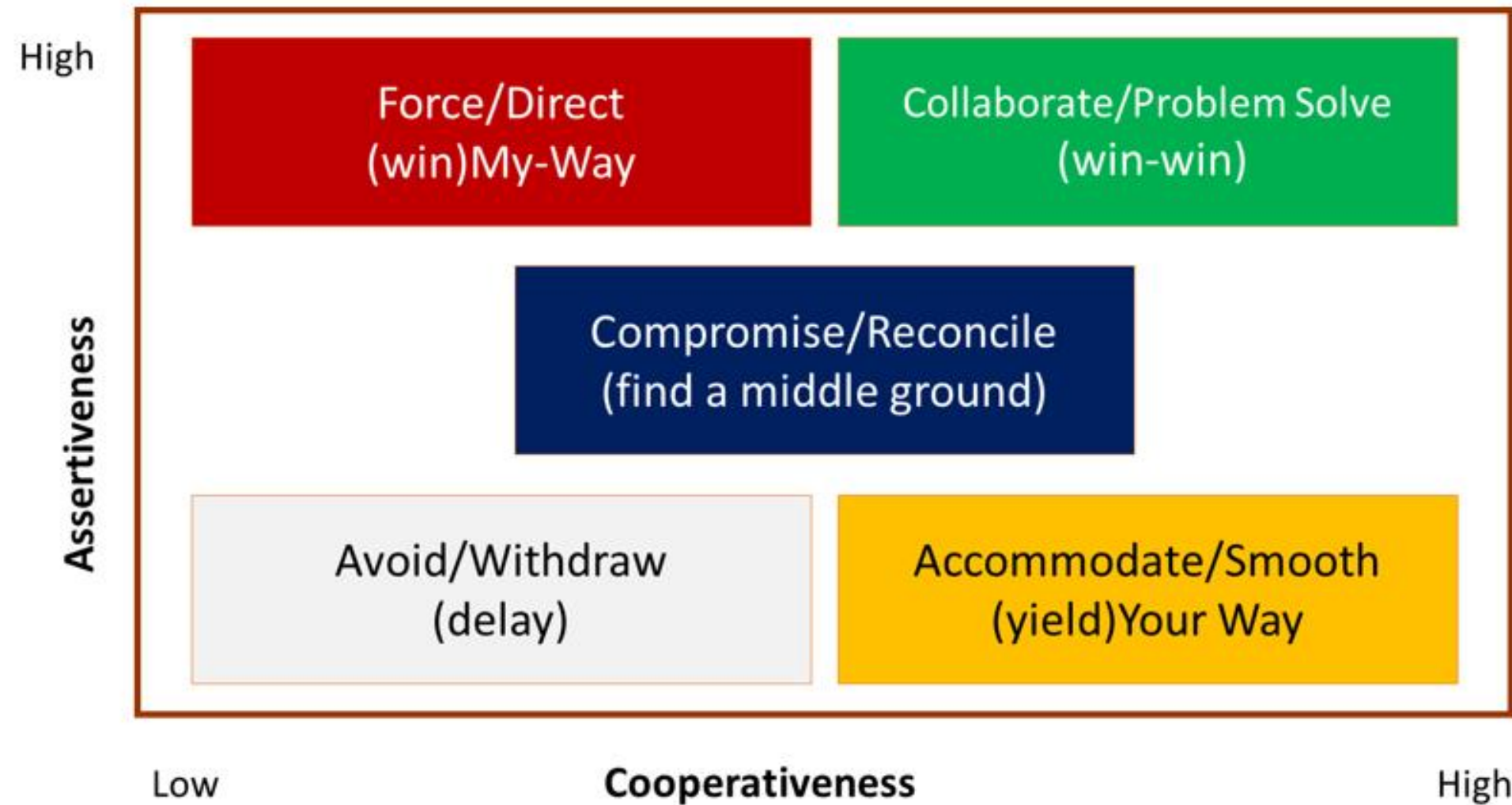
- Autocratic (A1):** You use the information that you already have to make the decision, without requiring any further input from your team.
- Autocratic (A2):** You consult your team to obtain specific information that you need, and then you make the final decision.
- Consultative (C1):** You inform your team of the situation and ask for members' opinions individually, but you don't bring the group together for a discussion. You make the final decision.
- Consultative (C2):** You get your team together for a group discussion about the issue and to seek their suggestions, but you still make the final decision by yourself.
- Collaborative (G2):** You work with your team to reach a group consensus. Your role is mostly facilitative, and you help team members to reach a decision that they all agree on.

The Vroom-Yetton Decision Tree: Adapted from Leadership and Decision Making by Victor H. Vroom and Philip W. Yetton by permission of the University of Pittsburgh Press. Copyright © 1973 University of Pittsburgh Press.



© Mind Tools Ltd 2018

Conflict Resolution Techniques



Skills and Knowledge



- Facilitation methods
- Participatory decision models (Convergent, shared collaboration)
- Principles of systems thinking (Complex, adaptive, chaos)





Including but not limited to:

- emotional intelligence
- collaboration
- adaptive leadership
- servant leadership
- negotiation
- conflict resolution

AGILE RETROSPECTIVES



The FEAR & VULNERABILITY Retrospective

ANDY CLEFF @andycleff

YOUR TOOLBOX

- With Developer Strengths
- Casual
- Kanban & Agile Model
- Missing Mentors

DEALING WITH YOUR FEARS

- Take responsibility for yourself
- Slow Down
- Confront your negative self-talk
- Explore your fear with conscious awareness



What are you AFRAID of?

How do we build a culture where we bring our whole selves + take off our masks?



TRUST + HONESTY

We want to be in the STRETCH ZONE...



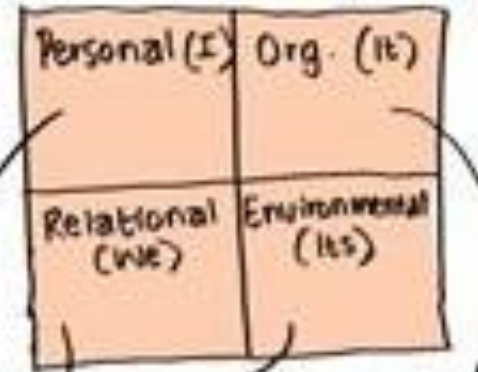
theme + dot vote

5 dots

what we see
TIME + TIME AGAIN



ACTIONS



I should ask for help when...

One thing we can both stop! Start is...

What I expect from my leadership is...

I'm proud of our team when...



SAFETY CHECK SURVEY

you can add to your board 25 a week team to work

Any format - here's some inspiration

Sketch by Julia @SketchingSA

courage **vulnerability**

resilience **Alliances**

Packs - Support Wild Caring inner Board Room

Steps



- Set the stage
- Gather data
- Generate Insights
- Decide what to do
- Close the retrospectives

Set the stage

FOCUS ON / FOCUS OFF

Inquiry rather than Advocacy

Dialogue rather than Debate

Conversation rather than Argument

Understanding rather than Defending

TYPES OF PARTICIPANTS

Explorer LHH

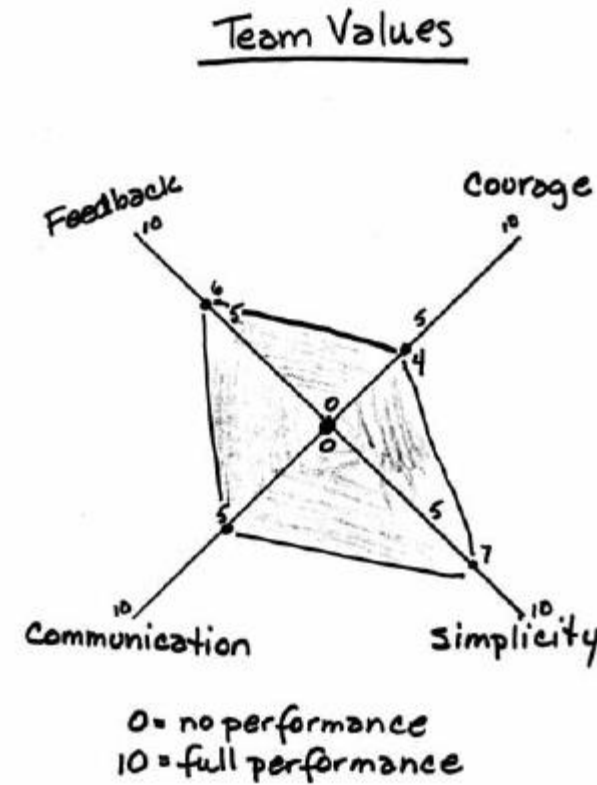
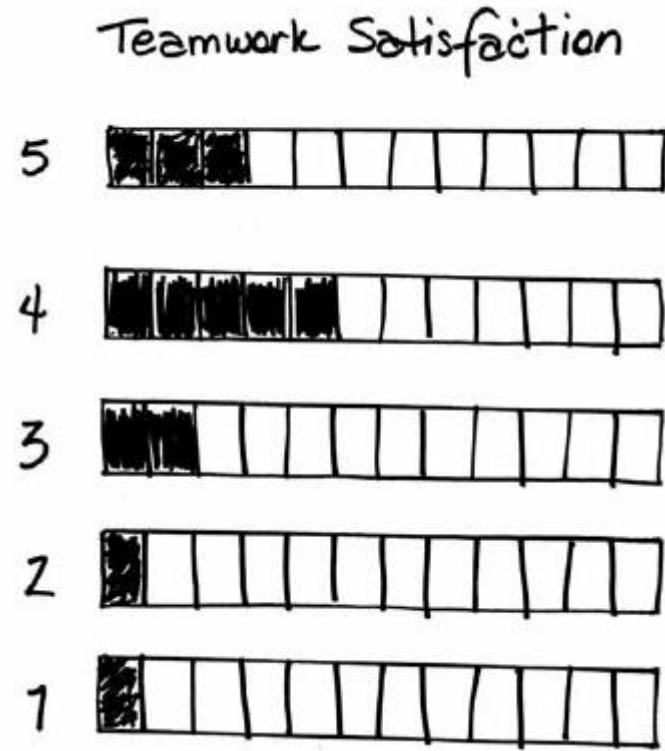
Shopper III

Vacationer I

Prisoner

Gather data

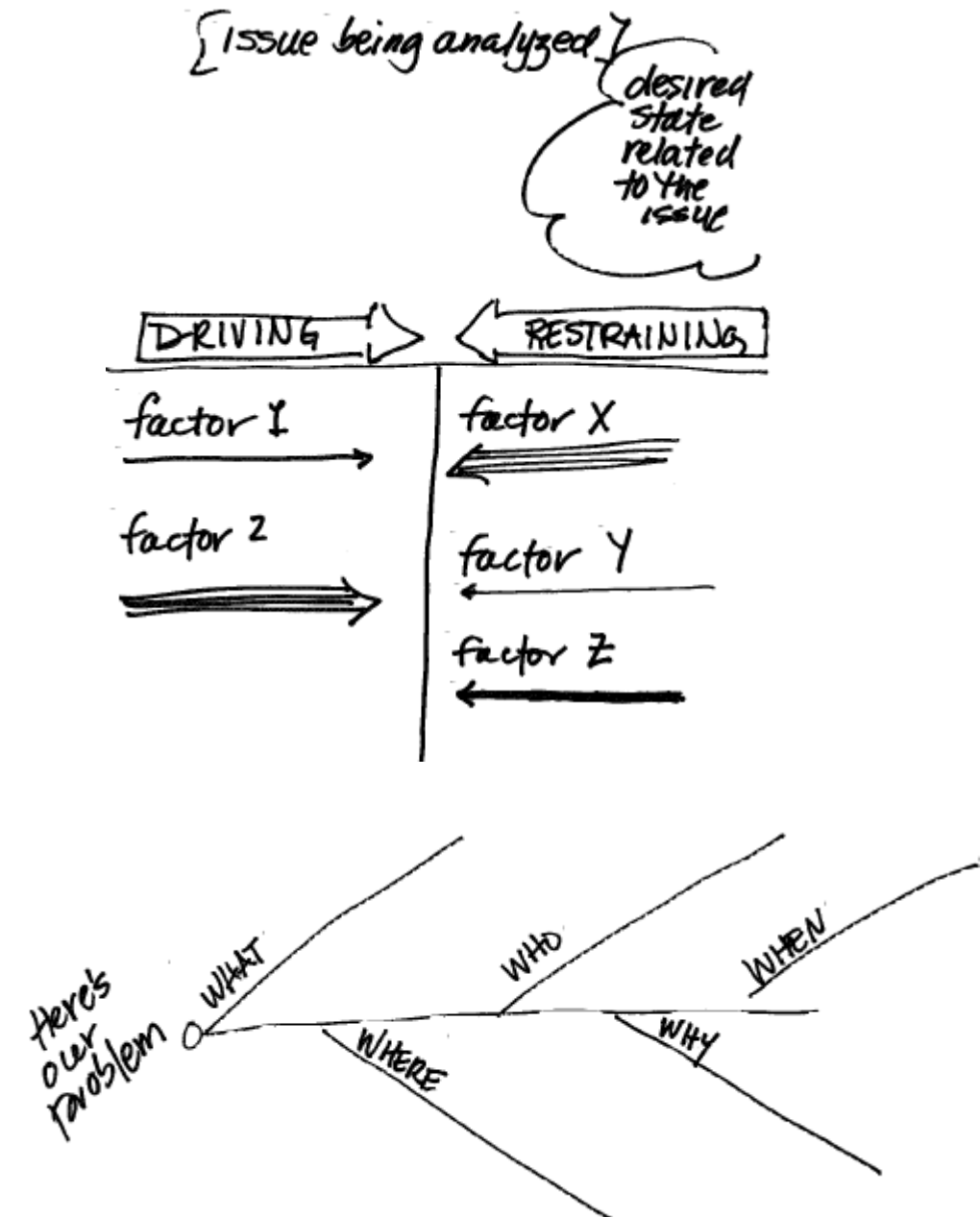
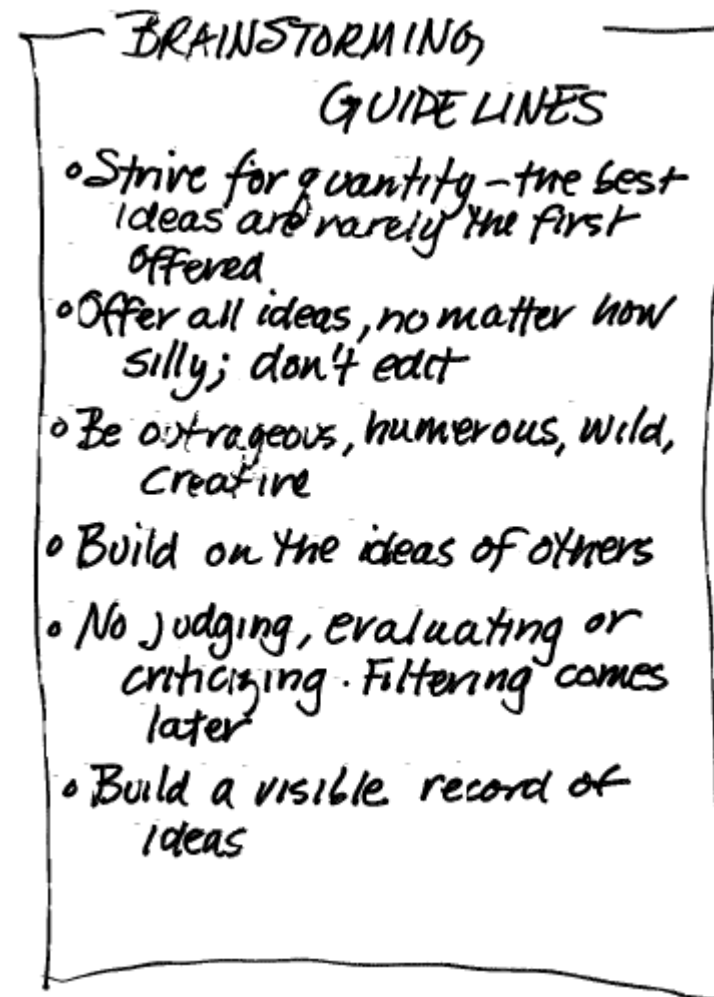
- Timeline
- Triple Nickels
- Color code dots
- Mad Sad Glad
- Locate strengths
- Satisfaction Histogram



- How Satisfied Are We?
Teamwork
- 5 = I think we are the best team on the planet! We work great together.
 - 4 = I am glad I'm a part of the team and satisfied with how our team works together.
 - 3 = I'm fairly satisfied. We work well together most of the time.
 - 2 = I have some moments of satisfaction, but not enough.
 - 1 = I'm unhappy and dissatisfied with our level of teamwork.

Generate Insights

- Brainstorming
- Fishbone
- 5 Whys
- Prioritize with Dots
- Learning Matrix



☺
 Kept to pairing schedule
 Velocity higher than ever
 Brown bag - refactoring to patterns
 Followed working agreement about feedback
 Kept the build going

☹
 Stayed late 3 nights
 Pair switching
 Bad snacks
 No celebrations

💡
 Invite other teams to brown bags.

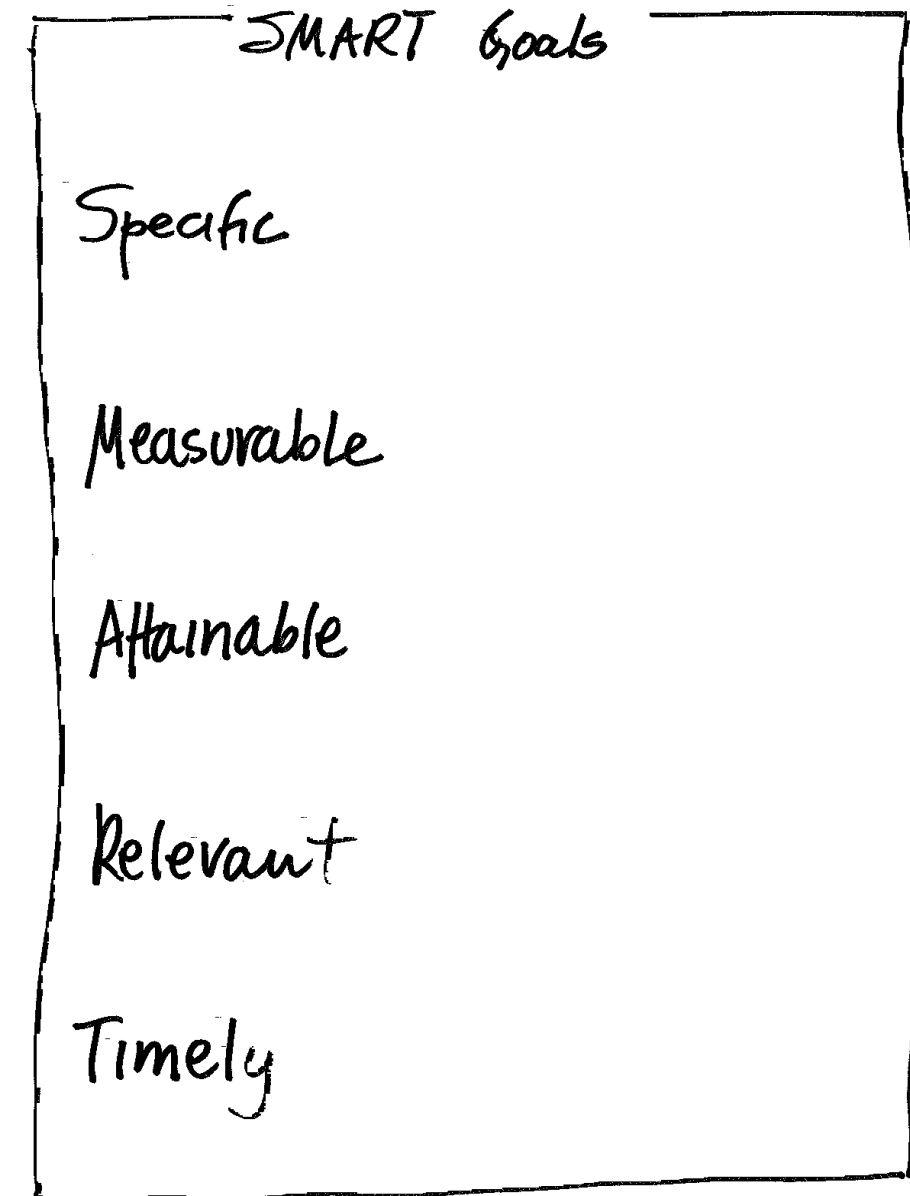
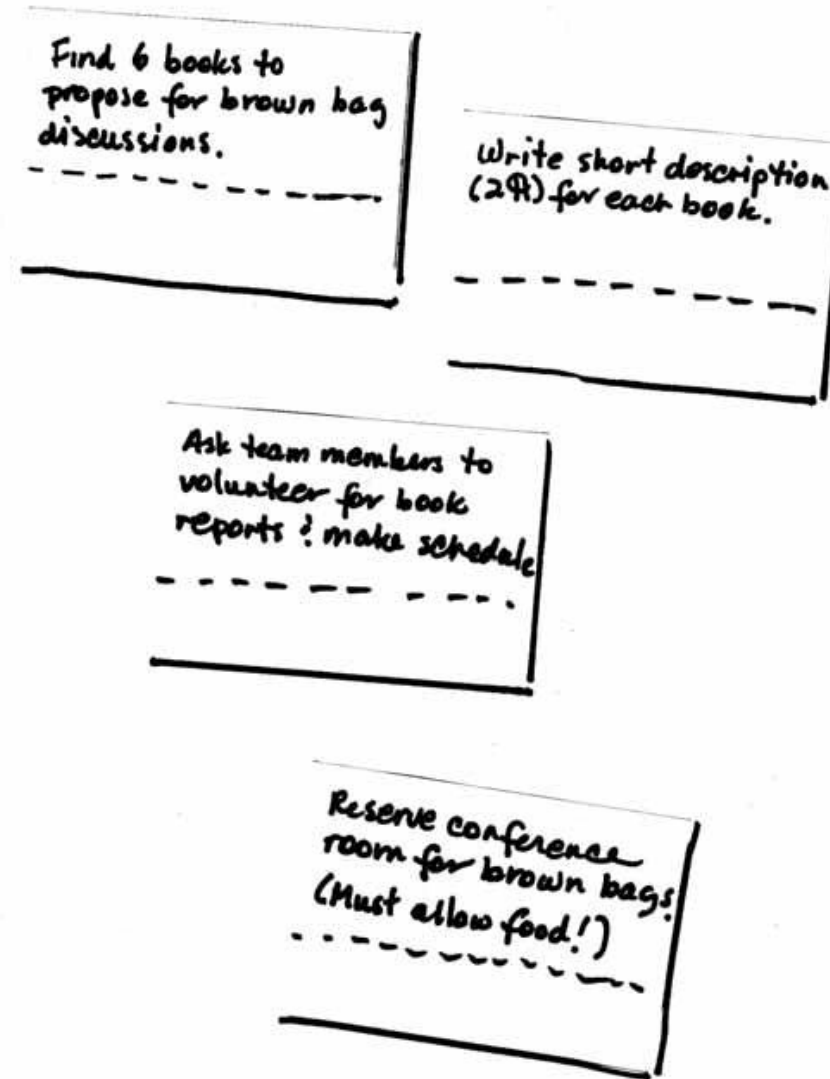
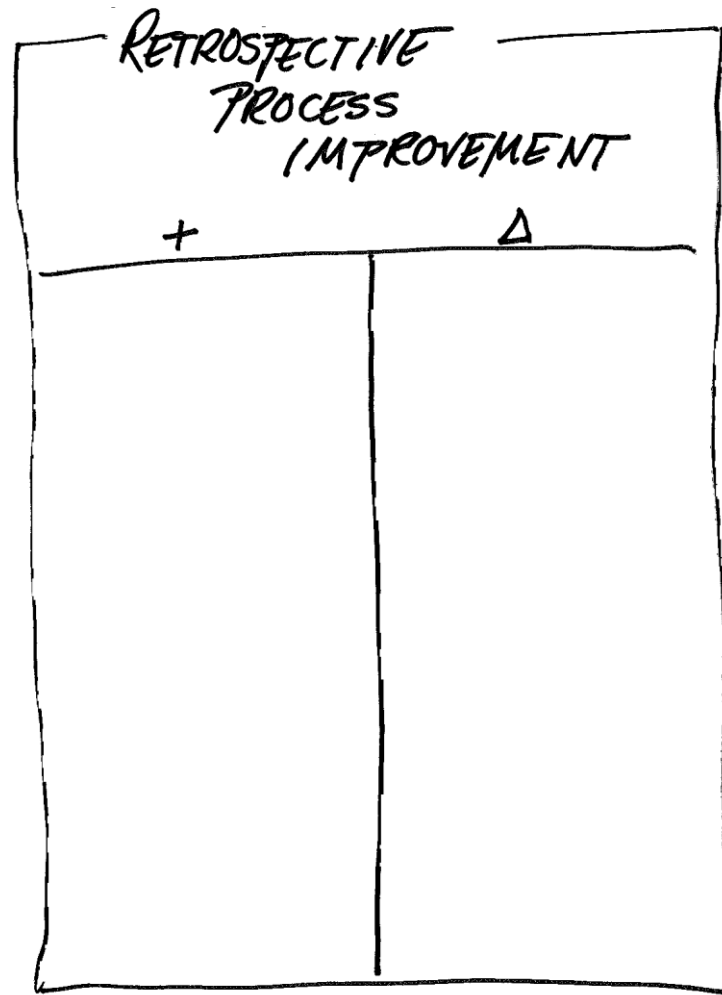
🌳
 Marco to Ulrike for brown bag
 Ulrike to Lisa for book recommendation
 Lisa to testing team for help with acceptance tests

Ideas for Team Experiments for Next Iteration

- Start brown bag - lunch & learn
- Increase pairing time to 5 hrs/day or 25 hrs/week
- Write more unit tests before coding
 - Measure time spent in "slack" activities
- Institute late penalty/fee for daily stand-up meetings
- Contact customer at least 2x a week
- More celebrations!
- More furniture for better communication flow
- More white board space

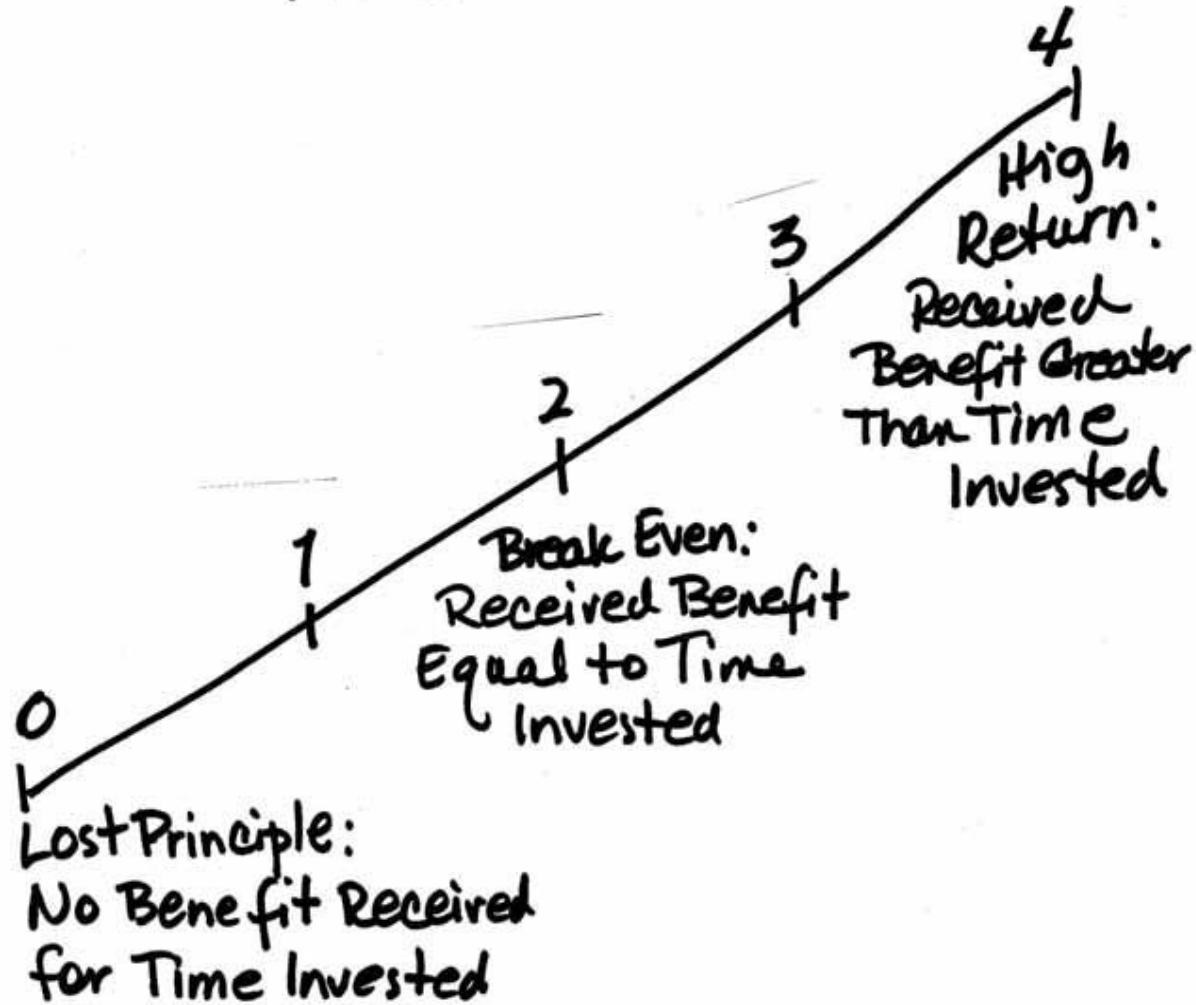
Decide what to do

- Retrospective planning game
- SMART goals
- +/-

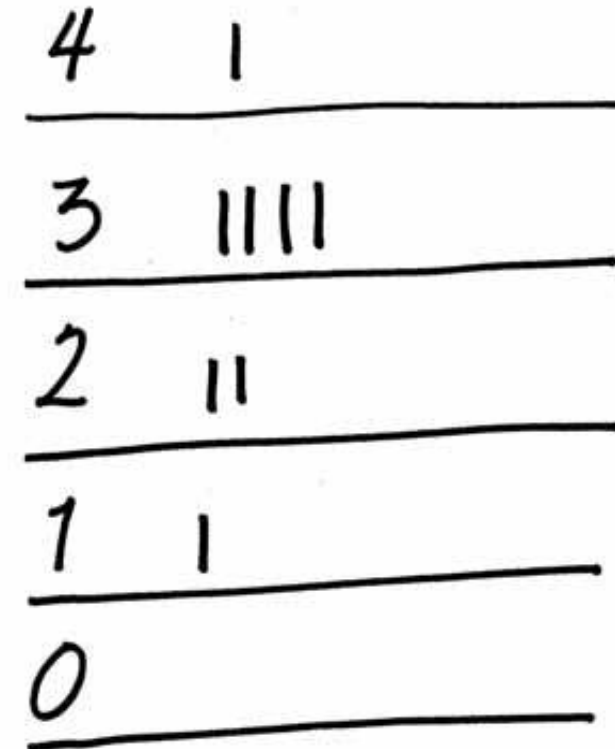


ROTI

Return On Time Invested



Our ROTI



Retrospective Minutes



Document



Group-M Retrospective Ceremony MarAprMay 2019 .msg



© Kaizen Institute

Process Improvement



Including but not limited to:

- Kaizen
- the Five WHYs
- retrospectives, introspective
- process tailoring/hybrid models
- value stream mapping
- control limits
- pre-mortem (rule setting, failure analysis)
- fishbone diagram analysis



<https://youtu.be/AbttByKuuhc?list=PLAyJQIy9UqoGdDZ59JgGHoy-ZYkCkr5Nd>

<https://www.youtube.com/watch?v=SrlYkx41wEE>

PMI Code of Ethics

PMI Code of Ethics and Compliance



PMI Code of Ethics



PMI Code of Conduct



- The PMI Code of Ethics and Professional Conduct outlines the following aspirational standards based on the core value of
- responsibility: base your decisions and actions on the best interests of society, public safety, and the environment;
- accept only assignments that are consistent with your background, experience, skills, and qualifications;
- honestly disclose gaps or weaknesses in your experience, qualifications, or skills;
- fulfill the commitments that you undertake – do what you say you will do;
- take ownership of errors or omissions and make corrections promptly;
- protect proprietary or confidential information that has been entrusted to you;
- Uphold the PMI Code and hold others accountable to it.



Mock Questions and Rapid-Fire Round





Appendix

Fun Acronyms of Agile –

<https://jessefewell.com/fun-acronyms-for-agile-software-development/>



- IKIWISI – I’ll Know It When I See It
- WILIWIK - What I like is what I know
- GEFN (pronounced, “gefen”) = “Good enough for now”
- WIIFM (pronounced “wif-im”) = “What’s in it for me.”
- WGLL (pronounced “wiggle”) = “What good looks like”
- WDLL (pronounced “widdle”) = “What Done Looks Like”
- WSTL (pronounced “whistle”) = “Who shouts the loudest”
- GASP (pronounced “gasp”) = “Generally Accepted Scrum Practice”
- “WSJF – Weighted Shortest Job First”
- CD3 – Cost of Delay Divided by Duration
- CRACK – “Committed, Responsible, Authorized, Collaborative & Knowledgeable” – Quality needed for the PO
- BRUF – Big Requirements UpFront

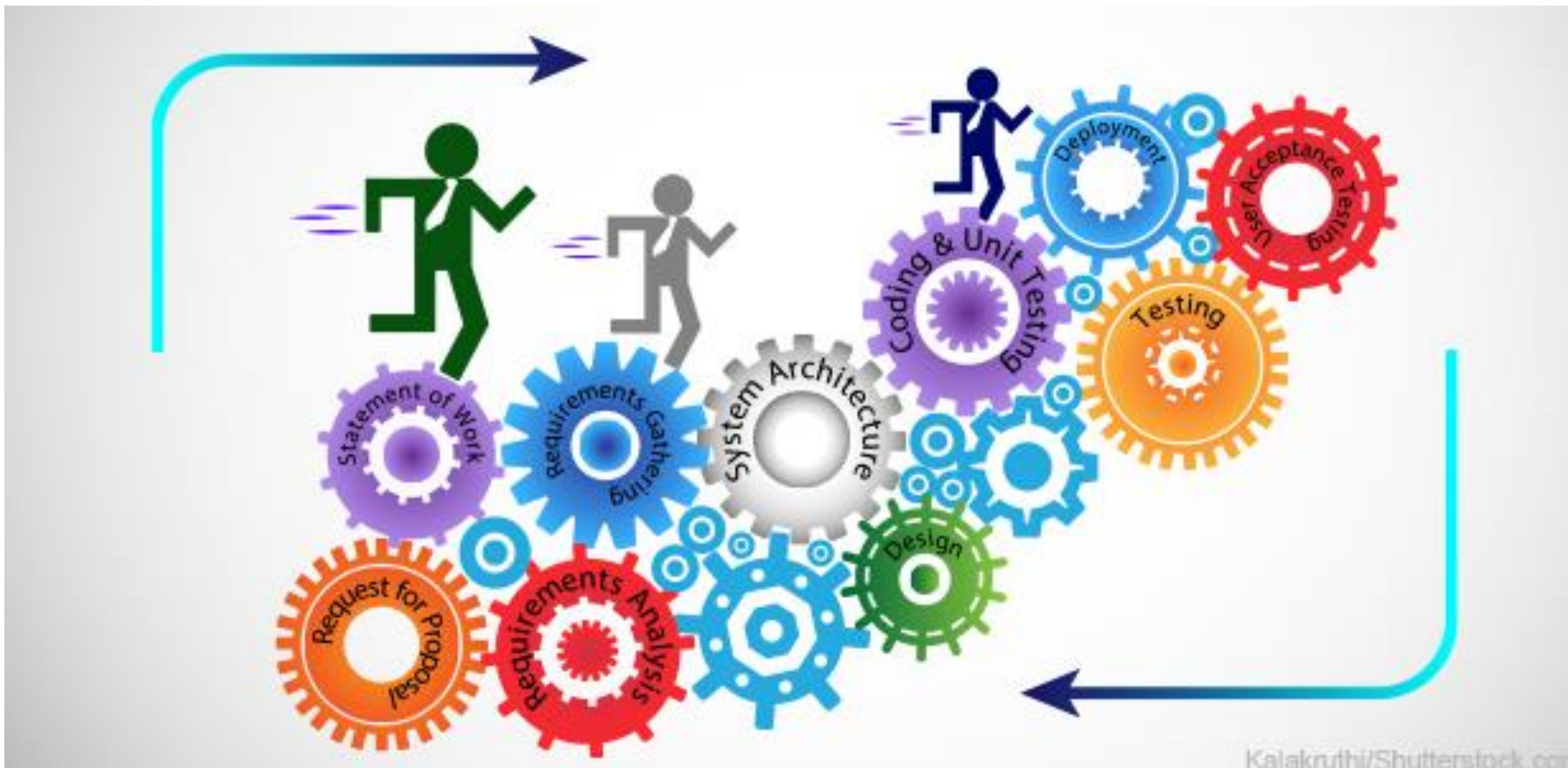
mnemonic

Key Abbreviations Explained!



- EVO – Evolutionary Model
- Spiral Model – Risk Driven Development
- IID – Iterative, Incremental Development
- DoD – Definition of Done
- RAD – Rapid Application Development
- RUP – Rational Unified Process
- FDD – Feature (Value) Driven Delivery

mnemonic





Benefits of **AGILE**

