ECE444: Software Engineering

RE 2: Requirements Elicitation, Persona

Shurui Zhou

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Learning Goals

- Basic proficiency in executing effective requirements interviews
- Understand that requirements are just “design data”, the information you will use to support your design
- Understand what/why/how about personas
- Recognize and resolve conflicts with priorities

Criteria

https://techtalkdotorg.files.wordpress.com/2015/02/requirements_analysis_focus.png

Inconsistency: Some requirements contradicts others.

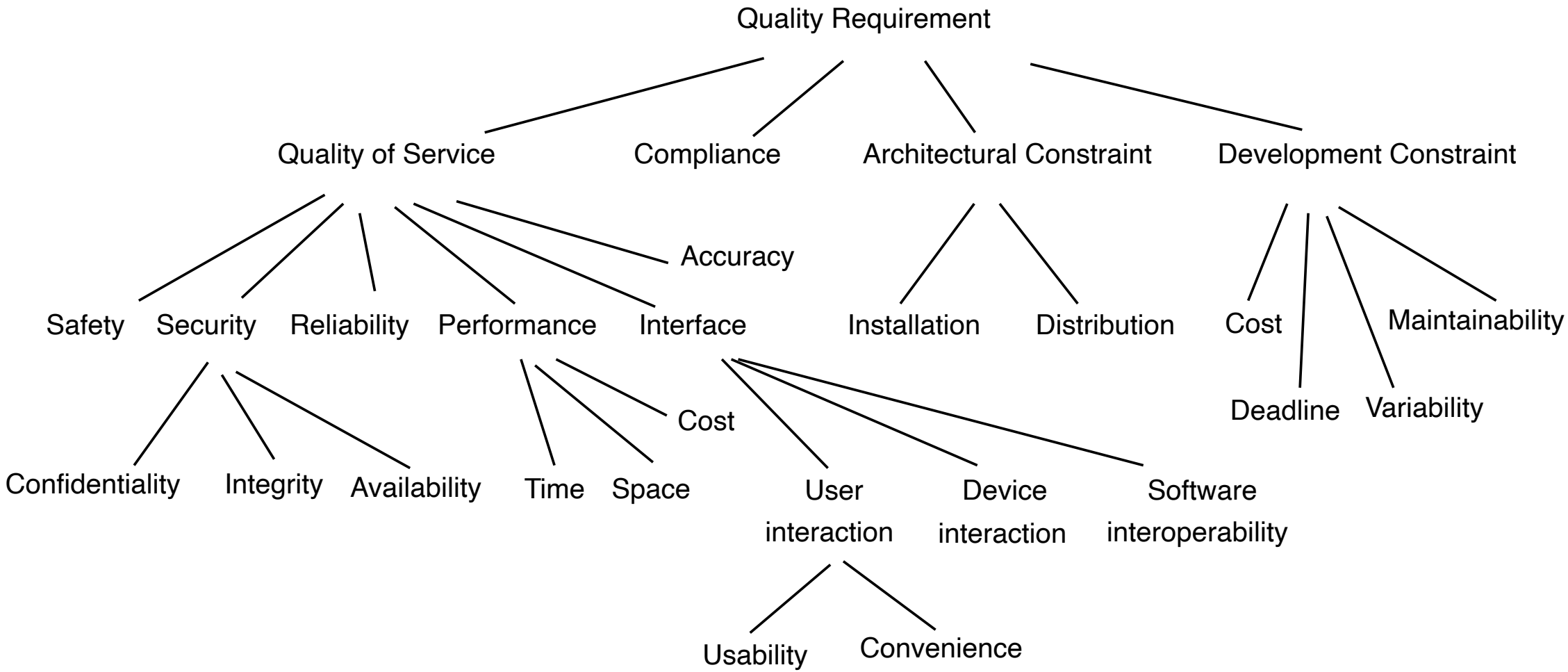
- One end-user wants that the heater be switched-off at the time the temperature of the heating element in the tower rises above some specified value, may be > 55°C
- Another end-user under similar circumstance may want the coolant circulation to be switched-on instead of heater being switched-off.

Incompleteness: Some requirements are omitted due to oversight.

- The analyst has not recorded: when temperature falls below 25°C, heater should be turned ON, coolant circulation should be turned OFF.

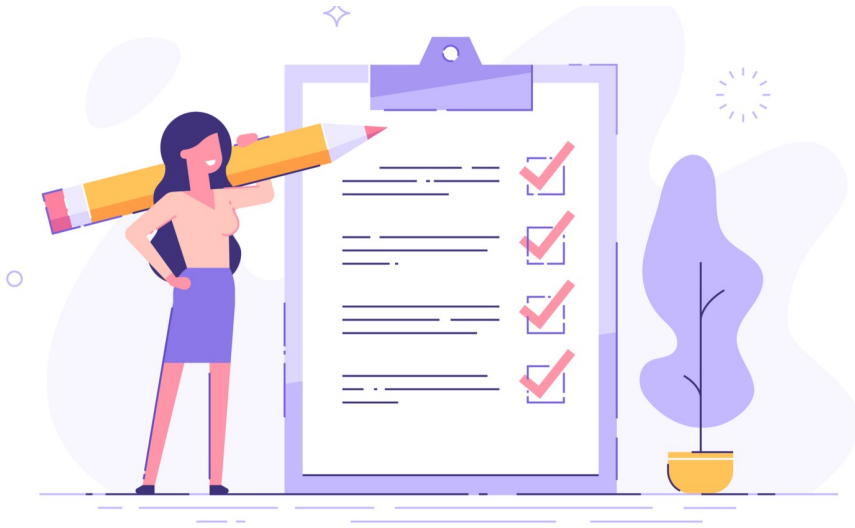
Anomaly: Is an ambiguity in requirement, several interpretations possible

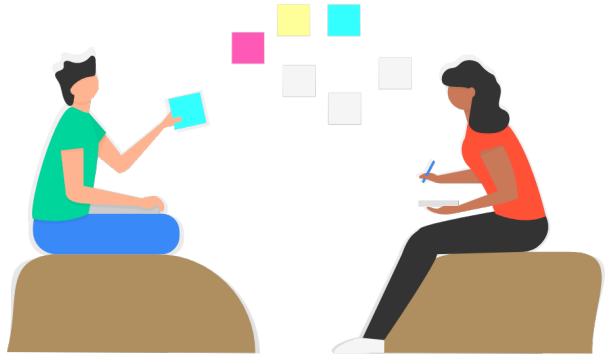
- When the temperature is high, the heater should be switched-off.


Quality Requirement
Quality of Service
Compliance
Architectural Constraint
Development Constraint
Accuracy
Safety
Security
Reliability
Performance
Interface
Installation
Distribution
Cost
Maintainability
Deadline
Variability
Confidentiality
Integrity
Availability
Cost
Time
Space
User interaction
Device interaction
Software interoperability
Usability
Convenience

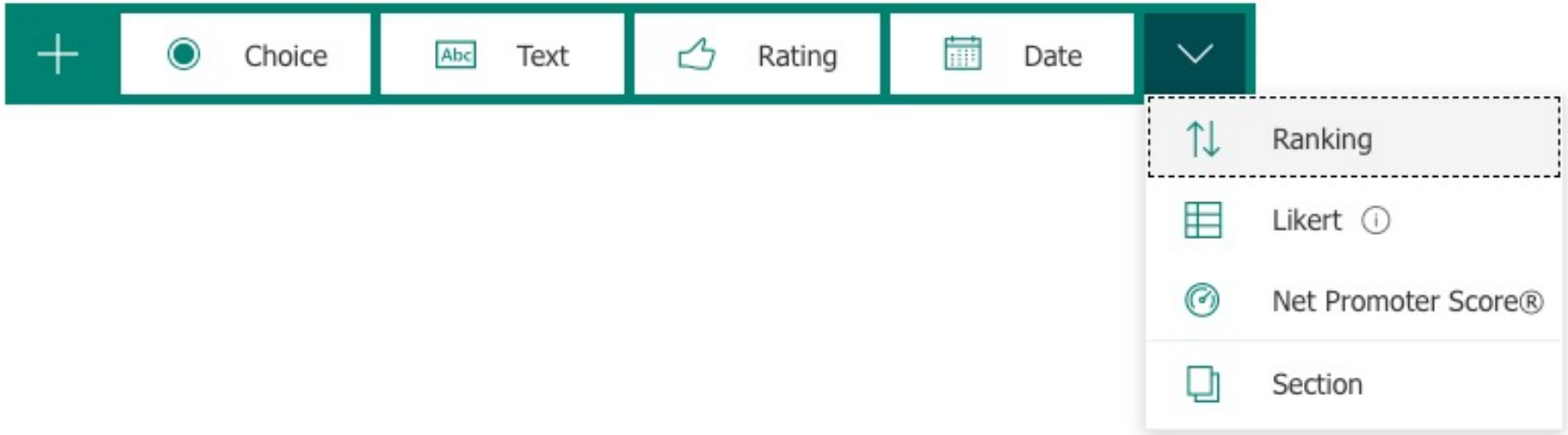
Collecting requirements: Elicit from stakeholders

- **Survey**: measure topics of interest in a controlled, consistent manner; easy to administer across large groups
  - Identify target population, their attitudes and preferences
  - Validate assumptions or facts
- **Interview:** More expensive, but could have follow-up questions to resolve ambiguity





Types of questions: depend on your goals



Interview

Interview Process

- Identify stakeholder of interest and target information to be gathered.
- Conduct interview.
  - (structured/unstructured, individual/group)
- Record + transcribe interview
- Report important findings.
- Check validity of report with interviewee.

About transcribing

- Transcribe the interview and identify the most relevant events to your project
- Do your best to follow the exact wording of the conversation. The key idea is to preserve any ambiguities and avoid summarizing or reinterpreting what the stakeholder says during the interview.
- Zoom has auto transcription; or exiting speech to text transcription tools (1) Temi (2) otter.ai
- Keep the transcription in your OneDrive

Kinds of questions

Opening questions: tell us who you are, where you work, and what you enjoy doing most outside of work

Introductory questions: introduce topic, what is the first thing that comes to mind when you hear ___ ?

Transition questions: think back to when… or, when does the process start?

Key questions: what is frustrating or useful about X? did anything change after using Y?

Ending questions: if you had a chance to change Z, what would you say? Did we miss anything?

Patton, M. Q. (1990). *Qualitative Evaluation and Research Methods, 2nded.* Newbury Park, CA: Sage Publications.

Sampling Strategies

- **Snowball/Convenience** – sample based on special access and proximity to investigator
- **Extreme/Deviant Case** – highly unusual, notable, exotic, top/bottom of topic
- **Typical/Common Case** – closest to centrality of the topic
- **Stratified Purposeful** – subgroups selected for comparisons
- **Maximum Variation** – illustrate dimensions of the topic to maximize variation

Interview Advice

- Get basic facts about the interviewee before (role, responsibilities, …)
- Review interview questions before interview
- Begin concretely with specific questions, proposals; work through prototype or scenario
  - Relate to current system, if applicable.
- Be open-minded; explore additional issues that arise naturally, but stay focused on the system.
- Contrast with current system/alternatives. Explore conflicts and priorities
- Plan for follow-up questions

Personas

Personas

"Personas are detailed descriptions of imaginary people constructed out of well-understood, highly specified data about real people"

—John Pruitt & Tamara Adlin

Partitioning the stakeholders into personas

Diversify your selections

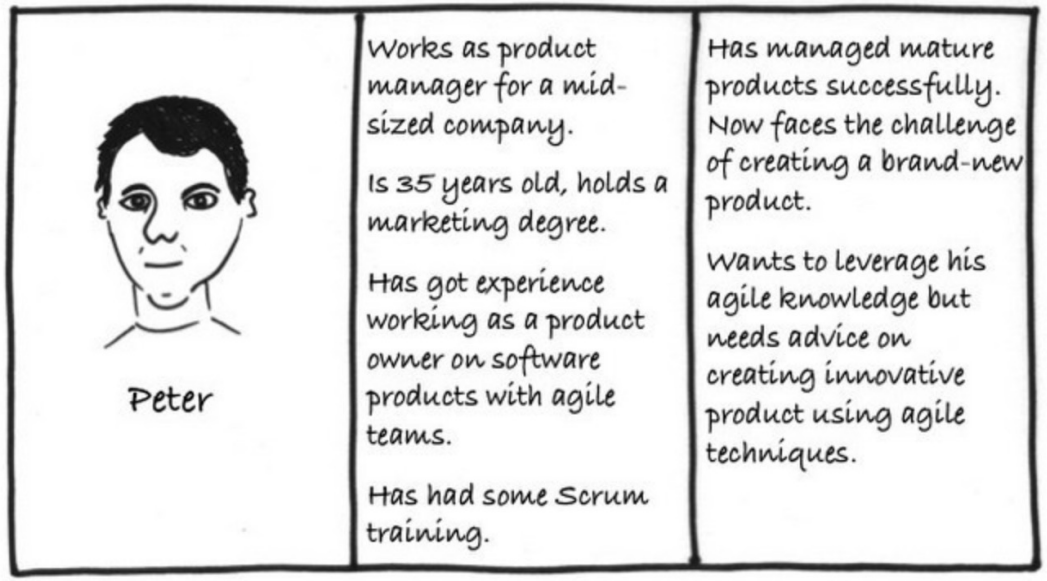
- The common case (most users)
- The extremes (rare, but demanding users)

Why create personas?

Personas…

- Guide developer decisions about features and how people interact with those features
- Help developers keep users and other stakeholders in mind during development
- Supplement (***but cannot replace***) developer access to stakeholders during iterations

Elements of a Persona



1.Persona Group (Banker, Hotelier, Web Manager)
2.Fictional name
3.Job titles and Major Responsibilities
4.Demographics (Age, Education, Ethnicity, and family status)
5.The goals and tasks they are trying to complete using the site
6.Their physical, social, and technological environment
7.A quote that sums up what matters most to the persona as it relates to your site
8.Casual pictures representing that user group

Running example: Time keeper

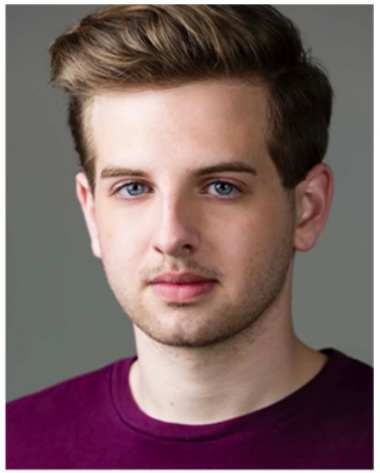
Project Description

This mobile application allows users to record and monitor how they use their time. The application tracks various types of activity, from work-related meetings to social events and outings. The application helps users achieve targets for increasing productivity and reducing stress through entertainment and social activities.

Project Objectives

- Enable users to track their activities throughout the day and balance work and life goals
- Integrate with existing calendaring and scheduling software
- Help users avoid missing deadlines and manage long-term project goals

Example Persona



Tom

Gender : Male

Age: 25

Status: In a relationship

Job: Graduate student & TA

Behavior & Belief

Tom lives in a rented apartment and commutes to college daily. He works as a T. A. and aims to get a good job, so that he can repay his student loan. He manages his tasks by writing down work in a calendar application. He stopped using time tracking software because he forgot to record his activities. He believes that quality of work is important and often spends huge amount of time doing one task. He mostly uses his laptop for his work and has an internet connection.

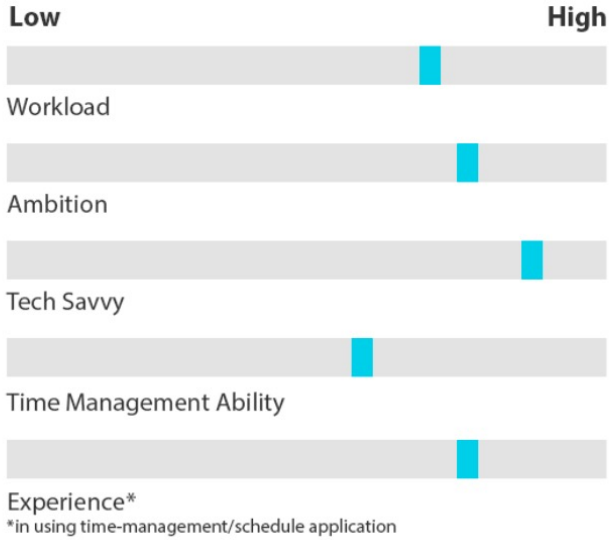
Goals

- To get a good job
- To repay his student loan
- To get good grades by finishing work on time

Values

- Automated tracking to record activities
- Analysis reports to better split time

Characteristics



Fears

- Forget to record his tasks

Pain Point

- Hates to manually enter time

"I would like to track my time but I often forget."

Example Persona

Lee

Gender : Male

Age: 23

Status: Single

Job: Graduate student, first year

Behavior & Belief

Lee had just started his first year in the MSE program. He is taking 51 units and is still getting the feel of how the workload is. But so far, he has been overwhelmed. He has had a lot of sleepless nights. He used the to-do list app on his phone but has not been using it since, Canvas has that feature. He has trouble focusing on a task for more than 30 minutes, so he does it over a period of time. Apart from his study, he practices vocal singing and plays badminton every day. He also likes to cook different dishes. He's on his mobile phone almost all the time to access the social media.

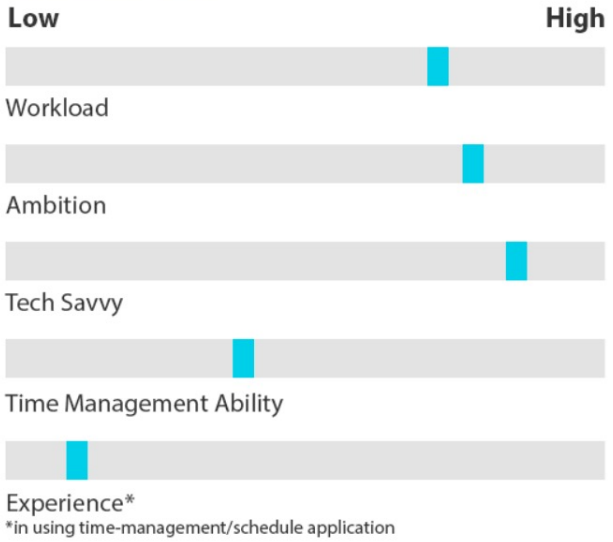
Goals

- Get a high GPA without burning out
- To continue following his hobbies

Values

- Get a reminder when he is behind schedule
- Get motivation to work
- Free to use

Characteristics



Fears

- Missing a deadline

Pain Point

- Managing deadlines

"My heart is in the work. It's hard to balance."

Example Persona

Cher

Gender : Female

Age: 35

Status: Married, one kid

Job: Full time employee
Part-time distance learning student

Behavior & Belief

Cher is 35 years old, married to Luke and is a mother of a 5 year old. She works as a business analyst in Chicago and is a distance learning student at University of Washington. She wakes up early to exercise, drops her kid to school, goes to work and studies while commuting and late in the evening. She uses multiple apps to balance her work and life. Even in her hectic schedule she sometimes manages to go for ice skating and dancing.

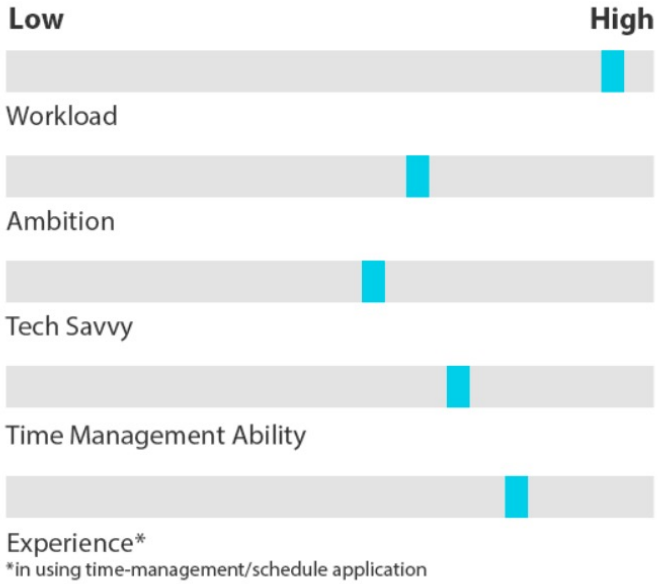
Goals

- To do well academically and advance in professional career
- No compromises on her family's well-being
- Continue to follow her hobbies
- To get good grades by finishing work on

Values

- Provide feedback on what could be done better
- Would prefer using only one app which fulfills her needs
- Distraction free

Characteristics



Fears

- Get delayed in important events such as picking up her kid from school

Pain Point

- Too many apps with redundant features

"Both family and work are very important. Plus, I also want to be a good student."

Running example: Chef Co-Pilot

The Gen-Z College Kid

Occupation
Student

Location
Toronto, ON

Age
21

Family
In a relationship

Device Usage

Browser

Mobile

Biography

Kevin is a typical Gen-Z college kid who rarely ever cooks at home. He lacks the time to think about his meals every day. Since he needs time to study, he cannot spend too much time on food. As a result, he tends to buy fast and unhealthy food. Even when he finds a recipe online that he likes, he often finds that he lacks most of the required ingredients. At most, he goes grocery shopping once a week. Being a college student, he also doesn't have too much money to spend on food.

Needs

- Have access to instructions which enable him to prepare fast and cheap meals with what he has at home.
- To make the best of his situation and eat somewhat healthy meals to keep up his mental stamina for school.
- To switch it up from the usual junk food he gets from nearby food trucks.

Pain points

Kevin needs a discovery platform which can recommend specific recipes based on his time and monetary restrictions, so he can cook more at home and eat healthier without investing too much effort.

The Main Cook of the House

Occupation
IT Manager

Location
Toronto, ON

Age
43

Family
Married w/ 2 kids

Device Usage

Browser

Mobile

Biography

John is a working parent who is responsible for cooking for his family of four, including his two kids. John's family eats dinner at 7 pm, around the time his kids get hungry and his partner gets home. With only an hour to cook dinner from when he gets home, John does not have the time to come up with new recipes and prefers making quick meals. John's grocery shopping occurs on a bi-weekly basis for which John needs to anticipate how much food is needed for the next two weeks.

Needs

- To have access to family-sized recipes that are quick & easy to make after a tiring day at work, taking <1hr to prepare.
- To buy enough food for two weeks of dinners to avoid grocery shopping too often.
- To make dinners that are healthy and enjoyable for his kids as well as the adults in the family.

Pain points

John needs a tracker and planner for the family meals. Cooking is supposed to be enjoyable, but not when it becomes a part-time job. He wants to spend less effort planning and cooking meals, and espcially not forgetting groceries.

The Foodie

Occupation
Software Engineer

Location
Seattle, WA

Age
25

Family
Married w/ no kids

Device Usage

Browser

Mobile

Biography

Jane loves eating food and trying new things. Eating for her is more than just satisfying hunger; it is an experience which should be enjoyed. To her, both health and taste are important. If time allowed, Jane would love to travel the world to experience all the amazing regional cuisines. What she really loves is reviewing different food items, sharing her opinions on social media, and learning the secrets of the tastiest dishes.

Needs

- To have access to chef-approved, restaurant quality recipes that can be made at home.
- Regardless of cost, to make aesthetically pleasing dishes for sharing.
- To have her opinions heard regarding how she thinks about the food she eats.

Pain points

Jane needs a community within which the best recipes can be shared and discovered. She needs a platform for different chefs & food experts to highlight the best recipes. Of course, being able to replicate and eat the best dishes is the dream.

The Healthy Professional

Occupation
Financial Advisor

Location
Toronto, ON

Age
32

Family
In a relationship

Device Usage

Browser

Mobile

Biography

Sherry is huge on keeping things organized and tracking her health. She is a working professional who spends a significant amount of time on other aspects of her life, such that managing her nutrition and fridge is not high on the priority list. When she wants something healthy after work, it's usually just a salad. When she wants to indulge in something sweet, she often finds nothing better in her fridge than old ice-cream.

Needs

- To keep track of what is in her fridge and be suggested what to purchase every time she visits the grocery store.
- To discover recipes based on her mood, such as for healthy foods and indulgent foods.
- To have a plan for her meals that incorporates a balanced diet that is healthy and tasty, and create that plan a week or even a month ahead of time.

Pain points

Sherry needs a personal "food advisor" that helps her manage her meals and scheduling, so she can focus on more important things in her life.

Creating Personas

Identify important categories of stakeholder

- Roles describe the kind of work people do, or their relationship in time to the product
- Goals describe what the users hope to achieve
- Segments describe shared demographic, attitudes or behaviors of your users

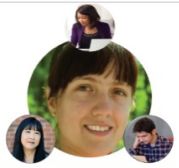
Exercise: Project 1

Education Pathways

Potential Personas?

The GenderMag Method

https://gendermag.org/custom_persona.php




The GenderMag Project


Home | GenderMag Method: What & How | The Personas | People & Sponsors | Flyers, Blogs, Webinars, & Presentations | Research Publications | GenderMag for Open Source

Abi (Abigail/Abishek)

Abi provides the strongest lens to find inclusiveness issues that disproportionately affect women users. If you choose to use only one persona and your primary motivation is inclusiveness to women, Abi is probably the best first choice. Abi provides the strongest inclusivity lens out of the 3 personas.

Tim (Timothy/Timara)

Tim represents the opposite end of the facet value ranges from Abi, so Tim helps to complete the "sweep" of considering a wide range of facet values. For full coverage of the spectrum, Tim is a good second choice after Abi.

Pat (Patricia/Patrick)

Pat touches (mostly) middle points in the facet ranges between Abi and Tim. If you want a third persona for additional coverage of the facets, Pat is a good third choice after Tim.

Abby Jones[1]

You can edit anything in blue print

- 28 years old
- Employed as an Accountant
- Lives in Cardiff, Wales

Abby has always liked music. When she is on her way to work in the morning, she listens to music that spans a wide variety of styles. But when she arrives at work, she turns it off, and begins her day by scanning all her emails first to get an overall picture before answering any of them. (This extra pass takes time but seems worth it.) Some nights she exercises or stretches, and sometimes she likes to play computer puzzle games like Sudoku

Background and skills

Abby works as an accountant. She is comfortable with the technologies she uses regularly, but she just moved to this employer 1 week ago, and their software systems are new to her.

Abby says she's a "numbers person", but she has never taken any computer programming or IT systems classes. She likes Math and knows how to think with numbers She writes and edits spreadsheet formulas in her work.

In her free time, she also enjoys working with numbers and logic. She especially likes working out puzzles and puzzle games, either on paper or on the computer

Motivations and Attitudes

- ***Motivations***: Abby uses technologies to accomplish her tasks. She learns new technologies if and when she needs to, but prefers to use methods she is already familiar and comfortable with, to keep her focus on the tasks she cares about.
- ***Computer Self-Efficacy***: Abby has low confidence about doing unfamiliar computing tasks. If problems arise with her technology, she often blames herself for these problems. This affects whether and how she will persevere with a task if technology problems have arisen.
- ***Attitude toward Risk:*** Abby's life is a little complicated and she rarely has spare time. So she is risk averse about using unfamiliar technologies that might need her to spend extra time on them, even if the new features might be relevant. She instead performs tasks using familiar features, because they're more predictable about what she will get from them and how much time they will take.

How Abby Works with Information and Learns:

- ***Information Processing Style:*** Abby tends towards a *comprehensive information processing style* when she needs to more information. So, instead of acting upon the first option that seems promising, she gathers information comprehensively to try to form a complete understanding of the problem before trying to solve it. Thus, her style is "burst-y"; first she reads a lot, then she acts on it in a batch of activity.
- ***Learning: by Process vs. by Tinkering***: When learning new technology, Abby leans toward process-oriented learning, e.g., tutorials, step-by-step processes, wizards, online how-to videos, etc. She doesn't particularly like learning by tinkering with software (i.e., just trying out new features or commands to see what they do), but when she does tinker, it has positive effects on her understanding of the software.

[1]Abby represents users with motivations/attitudes and information/learning styles similar to hers. For data on females and males similar to and different from Abby, see http://eusesconsortium.org/gender/gender.php

http://gendermag.org

Resolving Conflicts

Conflict Identification

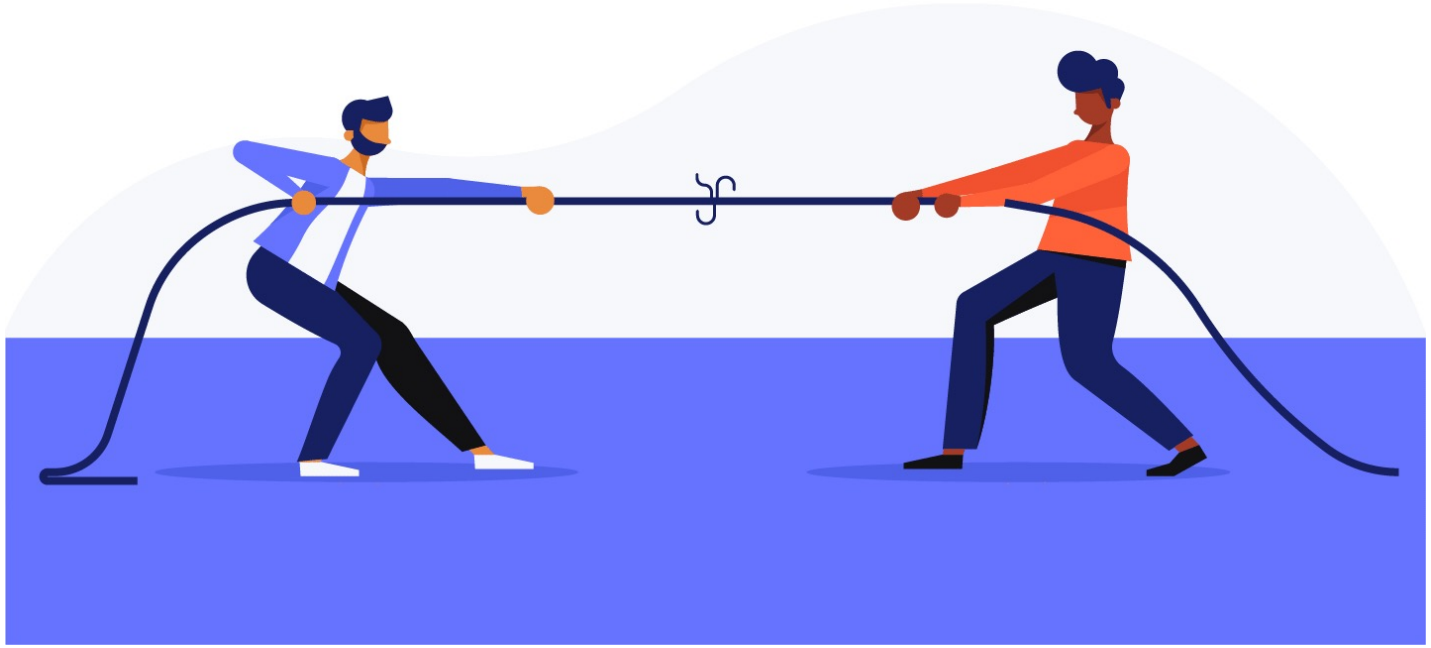
Example

***Human Resources** stakeholder group explicitly requests to capture the age of an employee,*

***Data Privacy team** is saying that the age of the employee may not be captured or used in reporting.*

Conflict Analysis

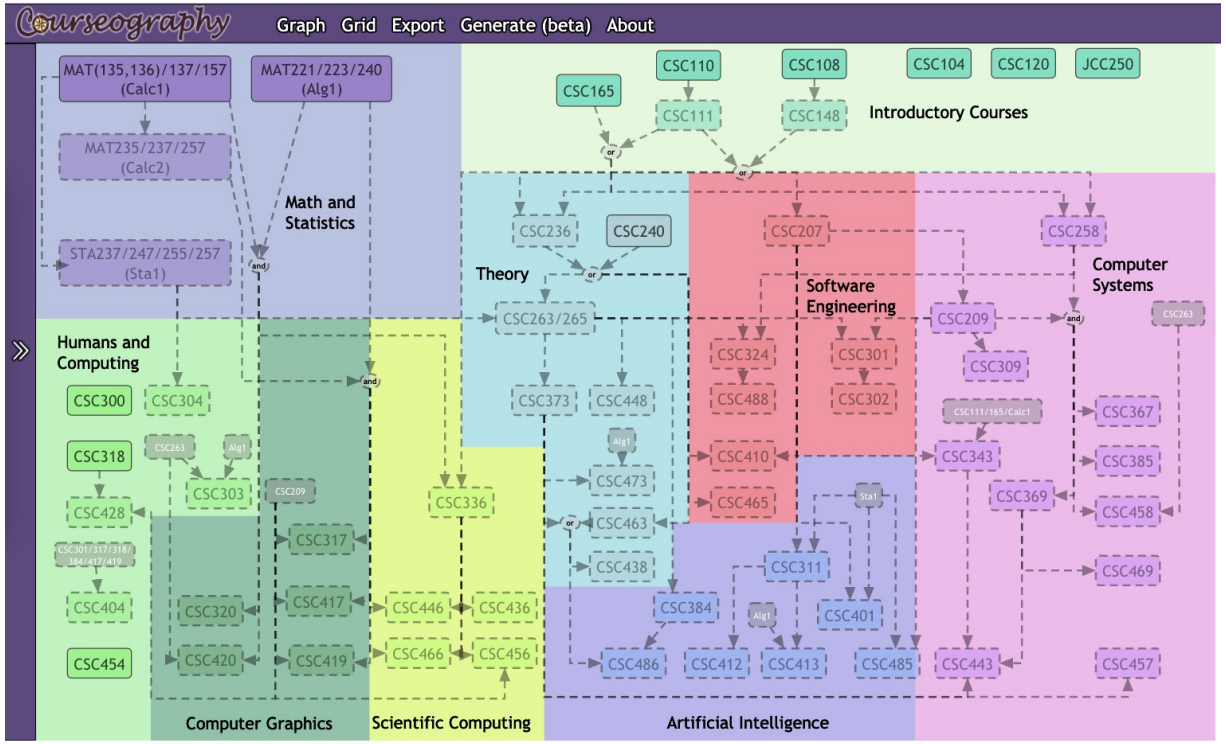
- Data Conflict
- Conflict of interest
- Conflict of Value
- Relationship conflict
- Structural conflict



Exercise: Project 1

Education Pathways

Any potential conflicts?



http://courseography.cdf.toronto.edu/graph

Magellan (online course selection tool)

Click here for ECE201's 2020- 2021 Magellan slides

Magellan is a software tool that has been developed in-house in order to help you plan and verify both your program and CEAB (Canadian Engineering Accreditation Board) requirements. It was developed in order to make it easy to verify the CEAB requirements as it automatically calculates the academic units (AU's) when you are building your study plans for 3rd and 4th year. In addition, it will confirm the program requirements at a glance.

MAGELLAN PRE-REGISTRATION (for *2021-2022 courses)* **ENDS: January 26, 2021 at 11:59pm EST**

Main profiles will then be *LOCKED until July, exact date TBC.* for preregistration planning.
ECE courses will be uploaded to ACORN timetables for students with **valid** Magellan profiles.
Magellan: http://magellan.ece.toronto.edu *(log in with UTORID and password)* / Magellan Manual

Main Profiles will be UNLOCKED in July
***CHANGES** made to main profiles once UNLOCKED in July*
*are **NOT** included in the preregistration process and thus not uploaded to ACORN.*

Types of inconsistency

- Terminology clash: same concept named differently in different statements
  - e.g. library management: “borrower” vs. “patron”
- Designation clash: same name for different concepts in different statements
  - e.g. “user” for “library user” vs. “library software user”
- Structure clash: same concept structured differently in different statements
  - e.g. “latest return date” as time point (e.g. Fri 5pm)

    vs. time interval (e.g. Friday)

Types of inconsistency, 2

- Strong conflict: statements not satisfiable together
  - e.g. “participant constraints may not be disclosed to anyone else”

    vs.

    “the meeting initiator should know participant constraints”
- Weak conflict (divergence): statements not satisfiable together under some boundary condition
  - “patrons shall return borrowed copies within X weeks”

    and

    “patrons shall keep borrowed copies as long as needed”

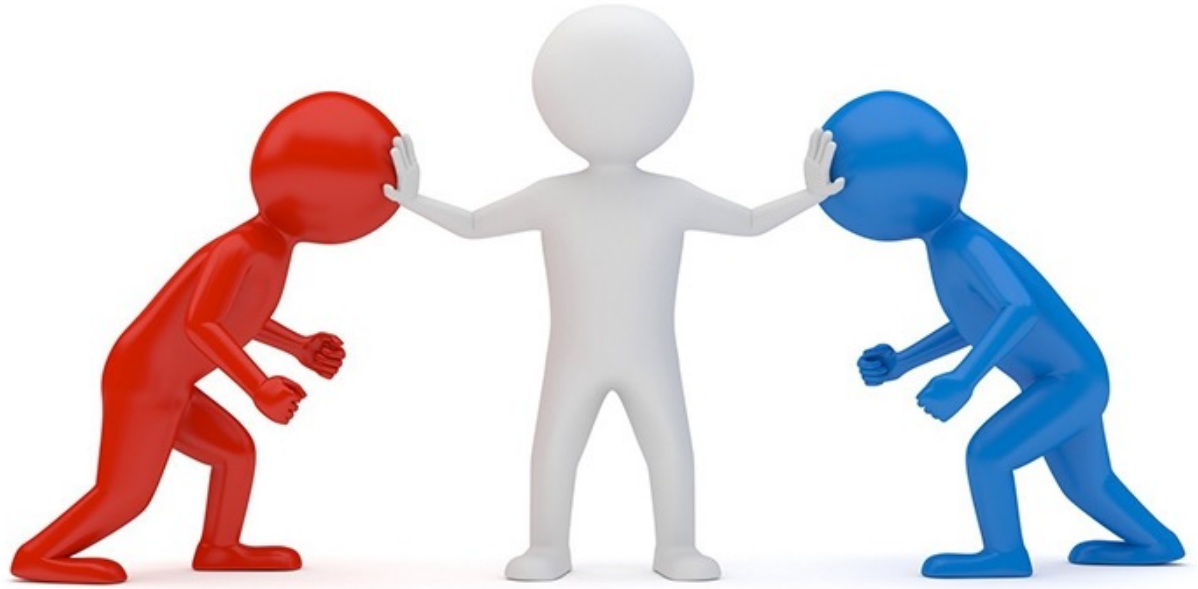
    contradict only if “needed>x weeks”

Handling inconsistencies

- Terminology, designation, structure: Build glossary, domain model
- Weak, strong conflicts: Negotiation required
  - Cause: different objectives of stakeholders => resolve outside of requirements
  - Cause: quality tradeoffs => explore preferences

Conflict Resolution

- Agreement
- Compromise
- Voting
- Definition of Variants
- Overruling
- …



Documentation of Conflict Resolution

- Document the complete detail of a conflict resolution to prevent the same conflict from arising again during the life of the project.

Requirements Traceability

- Keep connections between requirements
- What follows from what

Summary

- Many solicitation strategies, including document analysis, interviews
- Do not underestimate the challenge of interviews
- Resolving conflicts

Documenting Requirement

Many different forms

- Informal vs formal
- Unstructured vs structured
- Text vs diagrams
- Structured text common in practice
- Tool supported for traceability and process integration

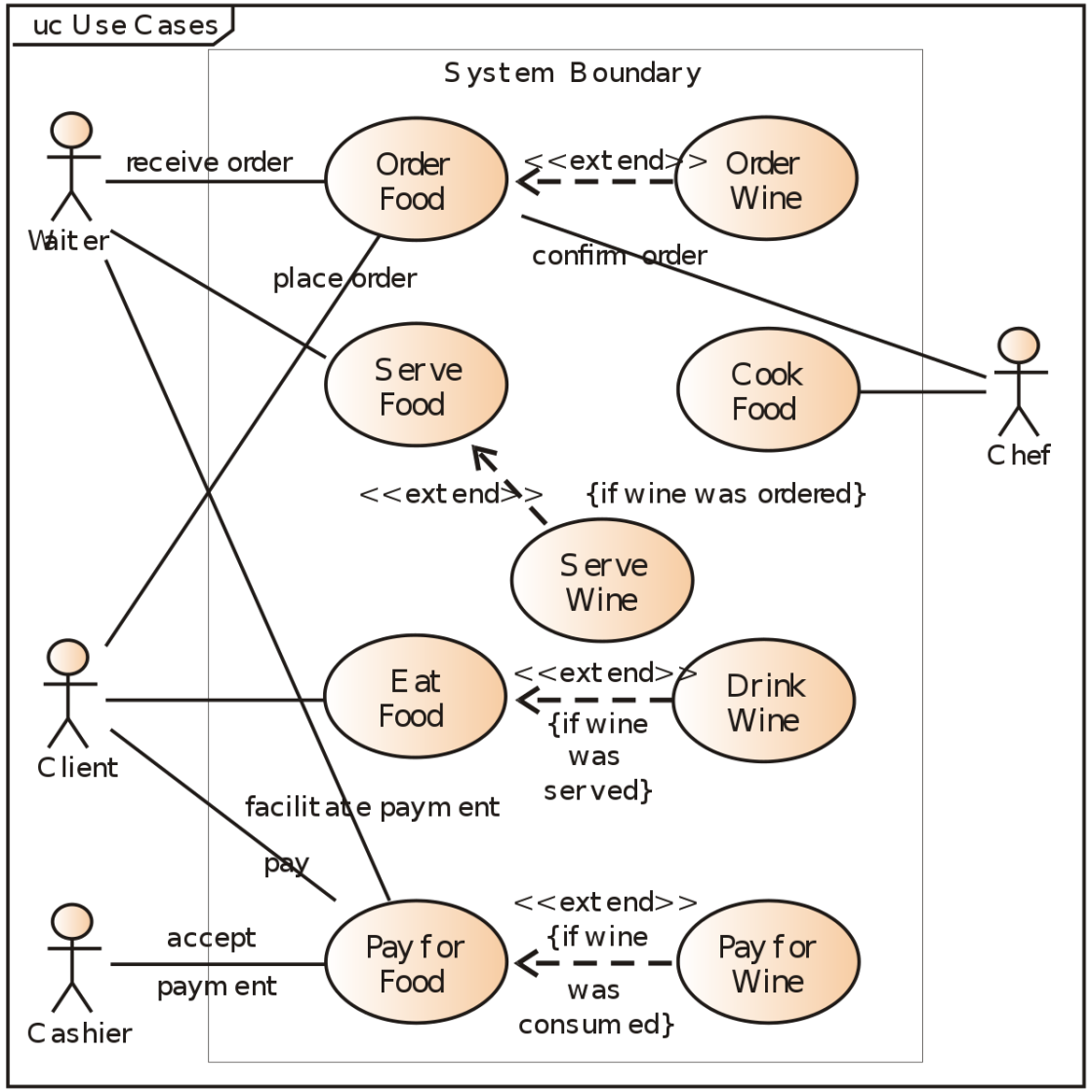
Software Requirements Specification (SRS)

- Formal requirements document
- Several standards exists
- Often basis for contracts

Table of Contents

Use Case Diagram

- Actor + action



Use Case

Use Cases help requirements analysts to…

- Identify actors and events around the system
- Define the system boundary –what is or is not within the system scope?
- Investigate early design interactions

(uses cases need not be descriptions of the final design)

Defining actors/agents

- An actor is an entity that interacts with the system for the purpose of completing an event [Jacobson, 1992].
  - Not as broad as stakeholders.
- Actors can be a user, an organization, a device, or an external system.

Use Case Templates

Use Case Name	Place order
Actors	(Primary) Store Manager, Sales Specialist
Pre-conditions	
Flow of events	
Post-conditions	

Use Case Templates

Use Case Name	Place order
Actors	(Primary) Store Manager, Sales Specialist
Pre-conditions	The store manager is under-stocked or the manager anticipates an increase in next period's sales
Flow of events	
Post-conditions	An order to restock the shelves is being processed

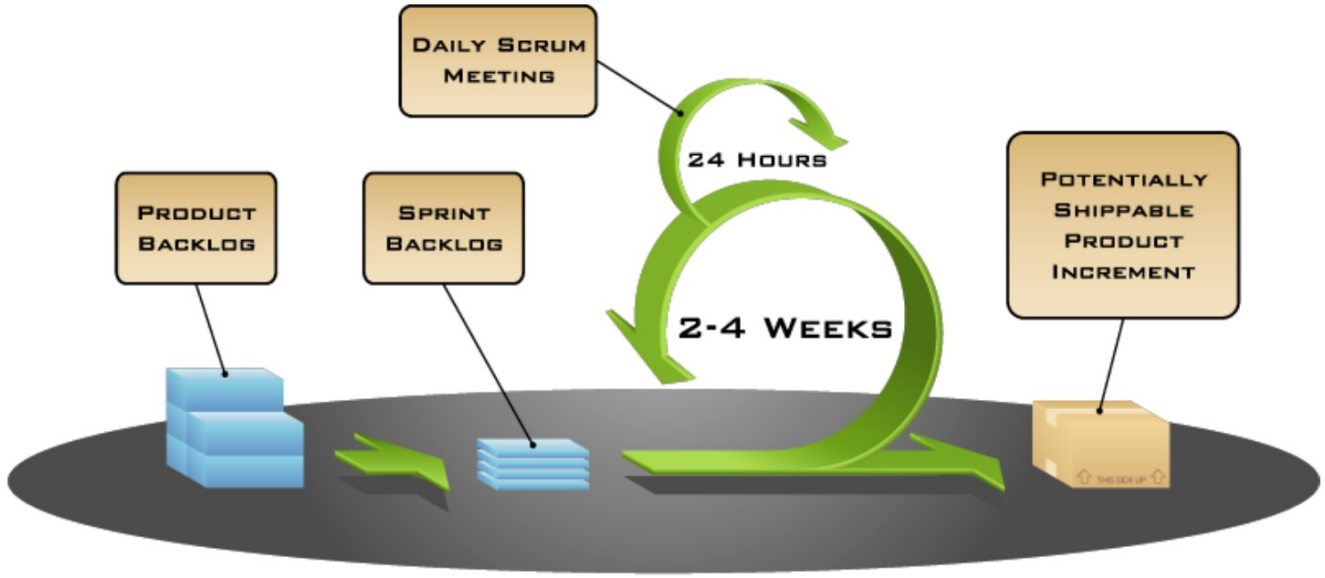
Use Case Templates

Use Case Name	**Place order**
Actors	(Primary) Store Manager, Sales Specialist
Pre-conditions	The store manager is under-stocked or the manager anticipates an increase in next period's sales
Flow of events	1. Sales specialist identifies manager's account 2. Manager finds the products to reorder 2.1 Manager browses or searches by keyword 2.2 Manager decides product quantities 3. Manager reviews and places the order 4. Specialist receives and processes the order
Post-conditions	An order to restock the shelves is being processed

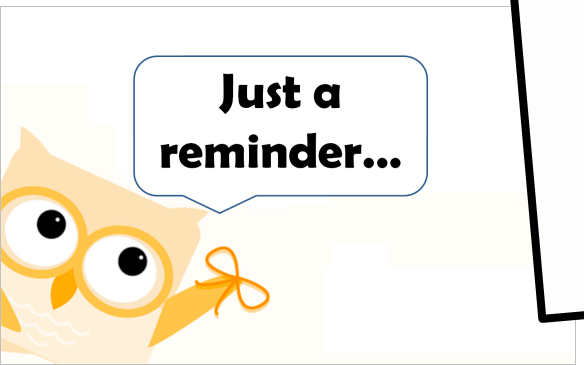
Use cases

Use Case Name	(Title)
Scope	System under design
Level	User level, subprocess level
Primary actor	(actors can be primary, supporting, or offstage)
Stakeholders, interests	Important! A use case should include everything necessary to satisfy the stakeholders' interests.
Preconditions	What must always be true before a scenario begins. Not tested; assumed. Don't fill with pointless noise.
Success guarantees.	Aka post conditions
Main success scenario	Basic flow, "happy path", typical flow. Defer all conditions to the extensions. Records steps: interaction between actors, a validation, a state change by the system.
Extensions	Aka alternate flows. Usually the majority of the text. Sometimes branches off into another use case.
Special requirements	Where the non-functional/quality requirements live.
Technology and data variations list	Unavoidable technology constraints; try to keep to I/O technologies.
Frequency of occurrence	
Miscellaneous	

Agile



Agile Development: User stories are the new requirements document



User Stories

- Common agile development practice
- Informal descriptions of user-valued features scheduled for implementation
- Details left for negotiation with customer later or pointer to real requirements
- A user story is a metaphor for the work being done, not a full description of the work.

User Stories

Who (User)

This should describe a fairly detailed user. It is not sufficient to just say "user." Strive towards something like "broke college student on a mobile device user." When we express the **who** with more detail we are able to better empathize with that particular user, determine the best solution and uncover implicit needs.

What (Goal)

The goal or action the user intends to take.

Why (Benefit)

Expressing the benefit to the user is by far the most important in my experience. Some of the most creative and inexpensive solutions come from the developers and users understanding why they are building something.

“persona + need + purpose”

- "As a [persona]": Who are we building this for? We’re not just after a job title, we’re after the persona of the person. Max. Our team should have a shared understanding of who Max is. We’ve hopefully interviewed plenty of Max’s. We understand how that person works, how they think and what they feel. We have empathy for Max.
- “Wants to”: Here we’re describing their intent — not the features they use. What is it they’re actually trying to achieve? This statement should be implementation free — if you’re describing any part of the UI and not what the user goal is you're missing the point.
- “So that”: how does their immediate desire to do something this fit into their bigger picture? What’s the overall benefit they’re trying to achieve? What is the big problem that needs solving?

User Story (Examples)

- As Max, I want to invite my friends, so we can enjoy this service together.
- As Sascha, I want to organize my work, so I can feel more in control.
- As a manager, I want to be able to understand my colleagues progress, so I can better report our success and failures.
- As UX Manager,

John wants centralized assets management

so that his designers are in sync.

- As a user, I can indicate folders not to backup so that my backup drive isn't filled up with things I don't need saved.

User Story (Examples) – how can we improve it?

- iPhone users need access to a vertical view of the live feed when using the mobile app.
- Desktop users need a “view fullscreen” button in the lower right hand corner of the video player.
- Android users need to be linked to apple store.

Exercise: Project 1

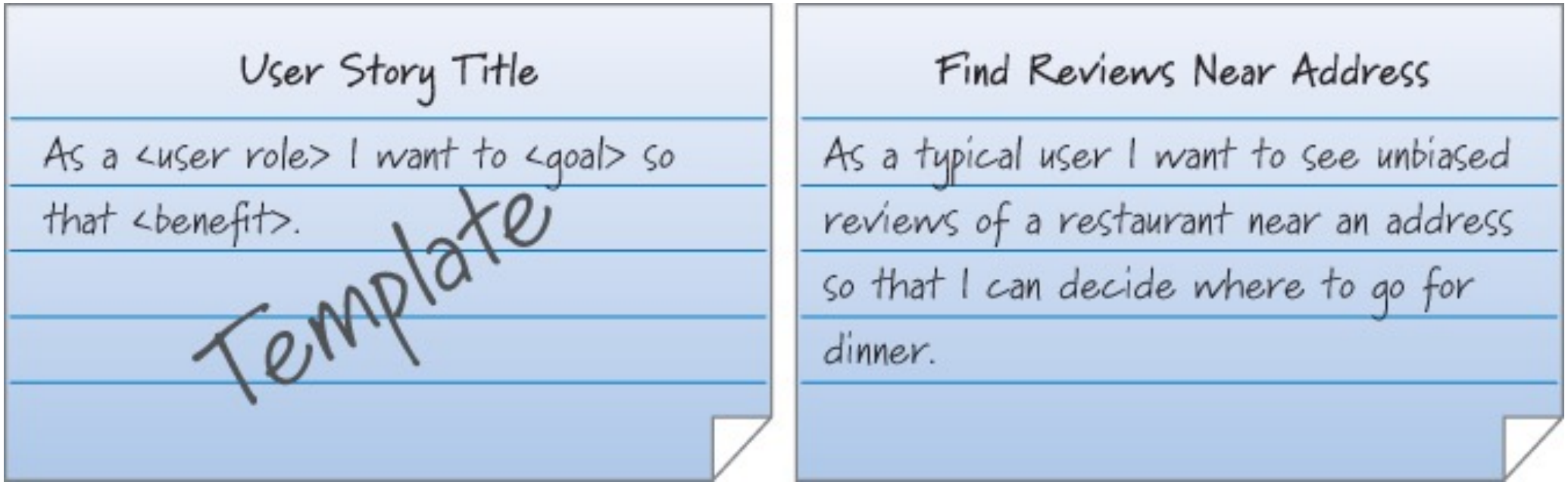
Education Pathways

a user story?

A Good User Stories -- Concept of 3C's

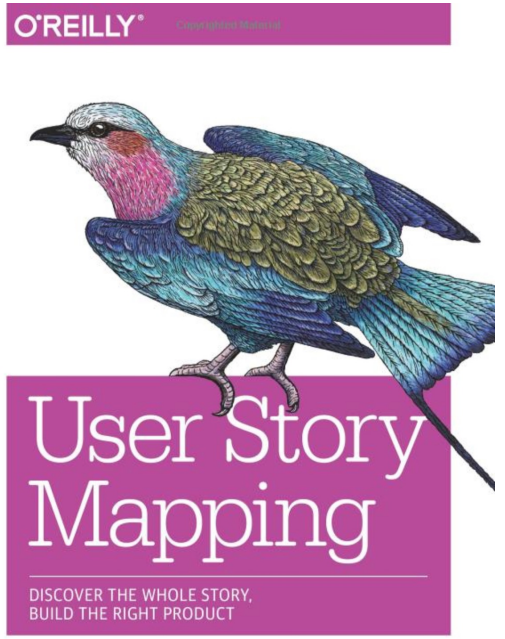
card	a brief, simple requirement statement from the perspective of the user
conversation	a story is an invitation for a conversation
confirmation	each story should have acceptance criteria

User Story Example - Card

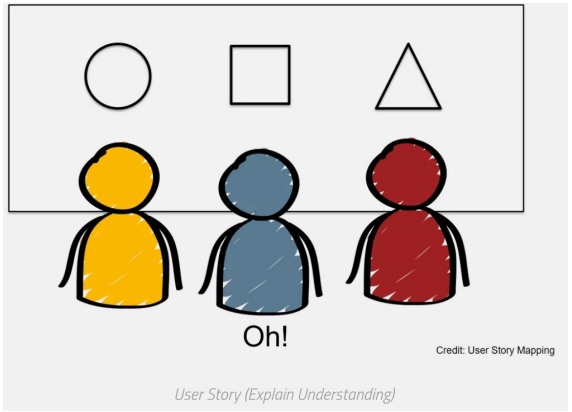


The conversation

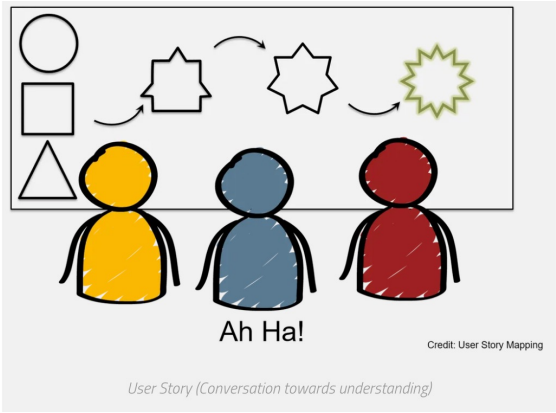
- An open dialog between everyone working on the project and the client
- Split up Epic Stories if needed



User Story (Not Shared Understanding)



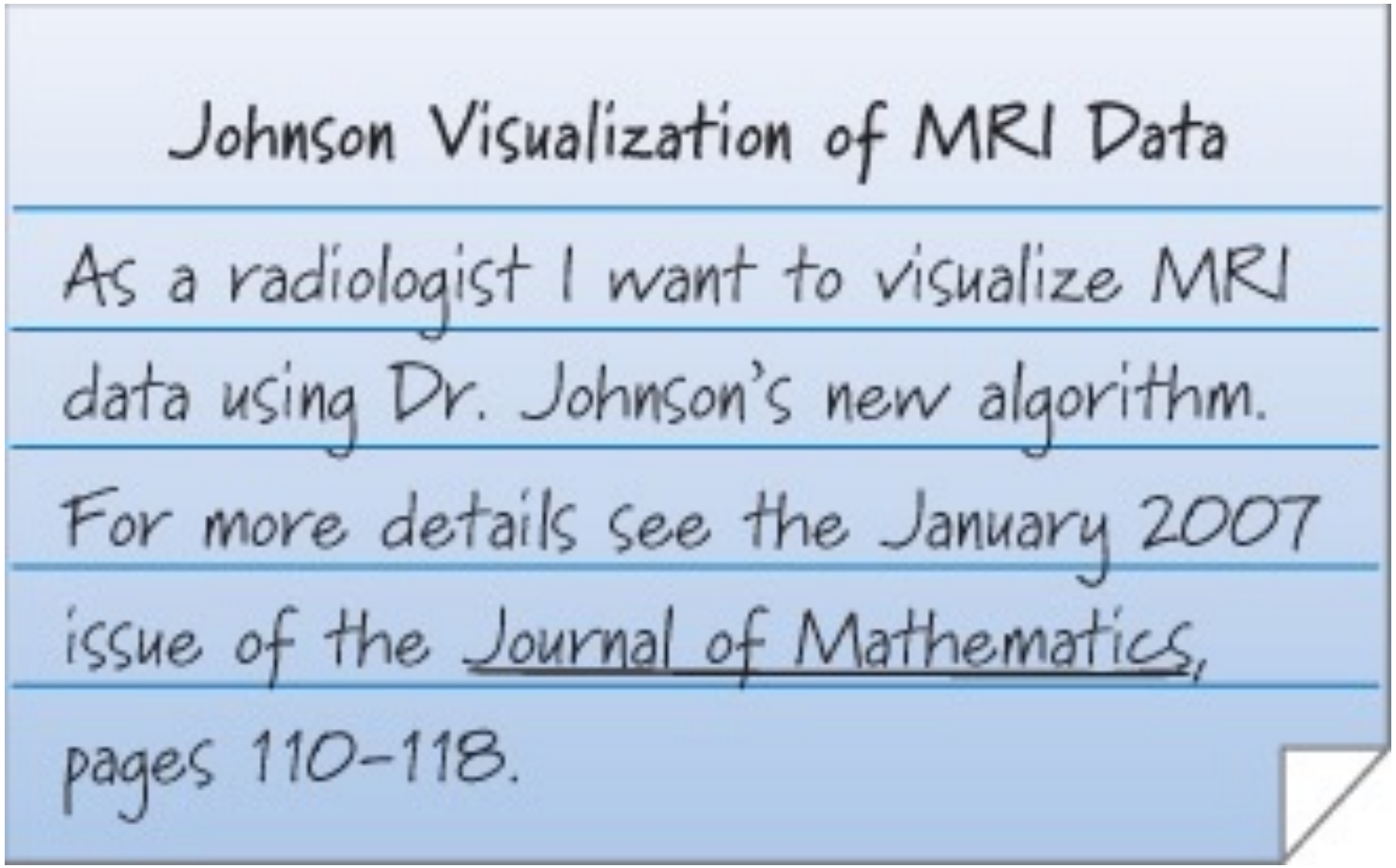
User Story (Explain Understanding)



User Story (Conversation towards understanding)

User Story (Shared Understanding)

User Story Example - Conversation

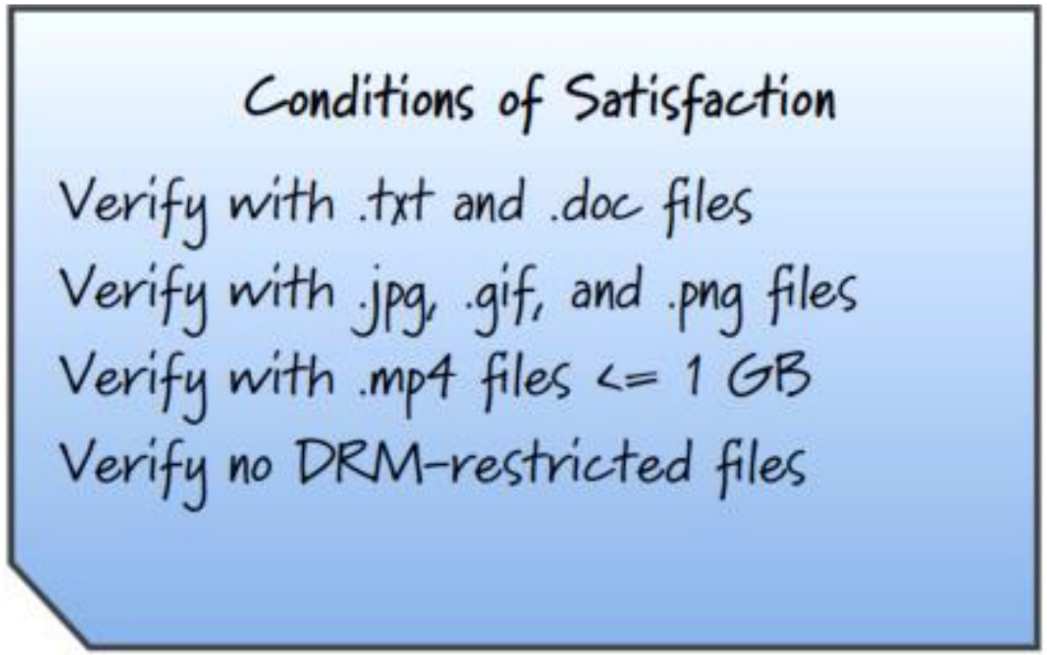


- Conversations may lead to a UI sketch, or an elaboration of business rules that gets written down.
- ← the user story references an entire article for future reading and conversation.
-

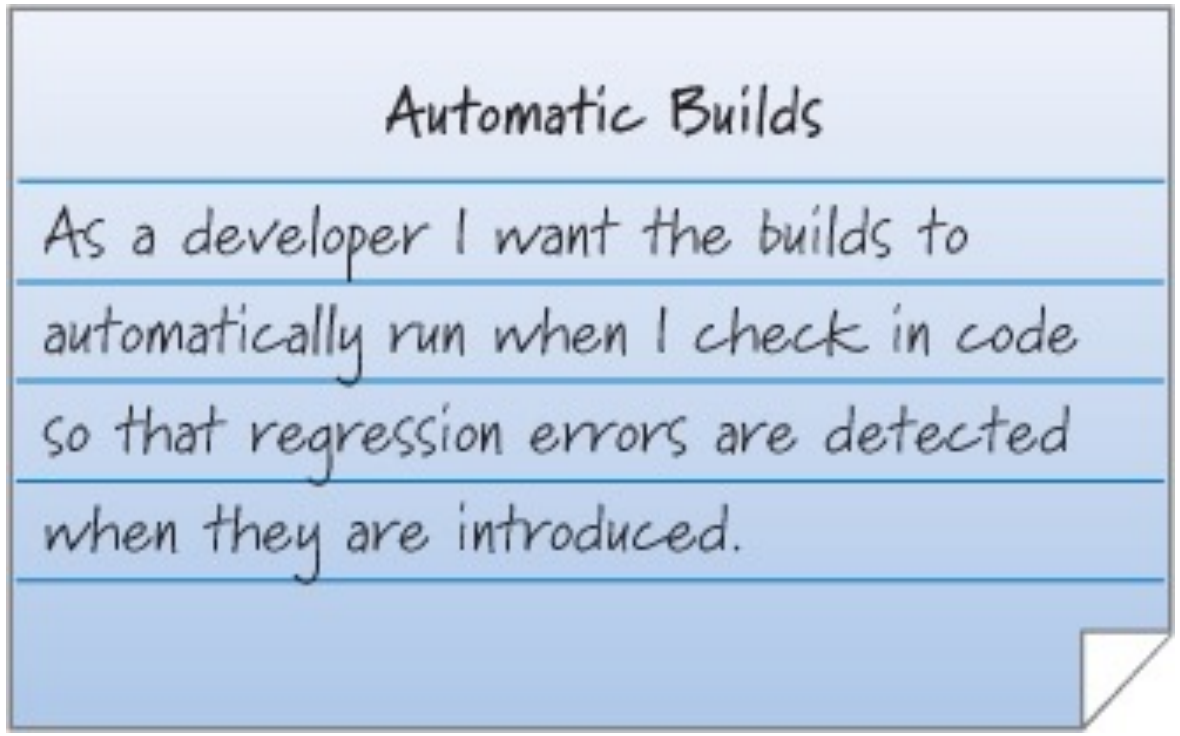
The Confirmation

- A confirmation criteria that will show when the task is completed
- Could be automated or manual

User Story Example - Confirmation



Technical Story



Non-Functional Requirements

- Security
- Performance
- Reliability
- Usability
- …

Some might be global, some local
– All responses should be below 3 seconds
– The wheel's revolutions per minute should be sampled 200 times per second to prevent aliasing effects

It is hard to reconcile global properties with agile principles

Non-Functional Requirements

Internationalization

As a user I want an interface in English, a Romance language, and a complex language so that there is high statistical likelihood that it will work in all 70 required languages.

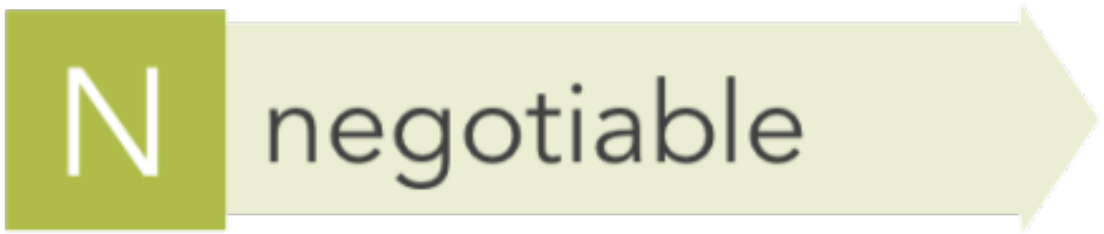
Web Browser Support

System must support IE8, IE9, Firefox 6, Firefox 7, Safari 5, and Chrome 15.

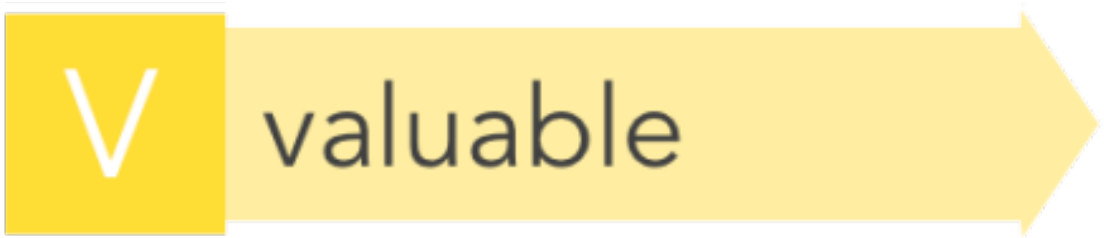
How to evaluate user study?

Follow the INVEST guidelines for good user stories!

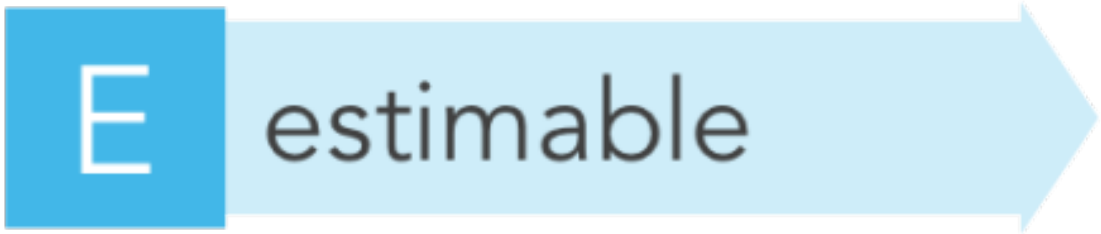
- Schedule in any order.
- Not overlapping in concept



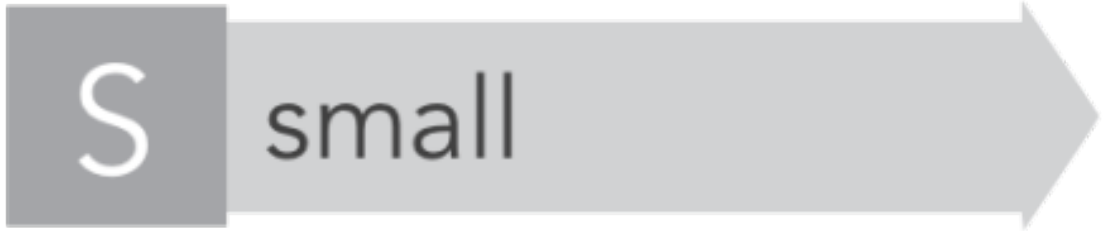
- A story is not a contract.
- A story is an invitation to a conversation.
- Details to be negotiated during development
- Good Story captures the essence, not the details



- If a story does not have discernable value it should not be done. Period.
- This story needs to have value to someone (hopefully the customer)
- Especially relevant to splitting up issues
- “so that ...”
- **Internal value** --- “non-functional requirements”



- Helps keep the size small
- A value with high value but extremely lengthy development time may not be the highest priority
- “Plans are nothing, planning is everything” -Dwight D. Eisenhower



- Fit on 3x5 card
- two week iterations -- user stories to average 3-4 days of work
- Too big == unable to estimate

- Ensures understanding of task
- We know when we can mark task “Done”
- Acceptance criteria can be written immediately.
- Unable to test == do not understand
- TDD

Agile Development: User stories are the new requirements document

Use cases

Use Case Name	(Title)
Scope	System under design
Level	User level, subprocess level
Primary actor	(actors can be primary, supporting, or offstage)
Stakeholders, interests	Important! A use case should include everything necessary to satisfy the stakeholders' interests.
Preconditions	What must always be true before a scenario begins. Not tested; assumed. Don't fill with pointless noise.
Success guarantees.	Aka post conditions
Main success scenario	Basic flow, "happy path", typical flow. Defer all conditions to the extensions. Records steps: interaction between actors, a validation, a state change by the system.
Extensions	Aka alternate flows. Usually the majority of the text. Sometimes branches off into another use case.
Special requirements	Where the non-functional/quality requirements live.
Technology and data variations list	Unavoidable technology constraints; try to keep to I/O technologies.
Frequency of occurrence	
Miscellaneous	

Is a User Story the same thing as a Use Case?

- **Not interchangeable**
- **User Stories** are centered on the result and the benefit of the thing you're describing
- **Use Cases** can be more granular, and describe how your system will act.

Use Cases vs User Story

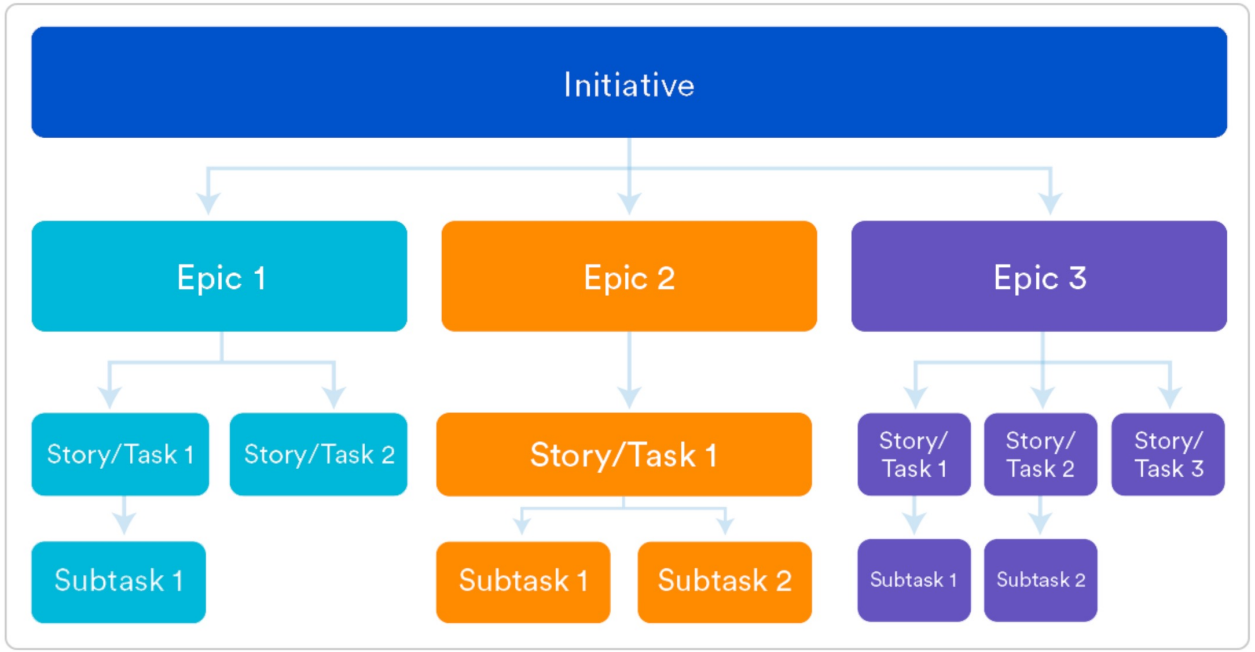
- Similarity
  - User Stories: user role, goal and acceptance criteria.
  - Use Cases: an actor, flow of events and post conditions
- Difference
  - Less details in User Story
  - Small increments for getting feedback more frequently, rather than having more detailed up-front requirement specification as in Use Cases.

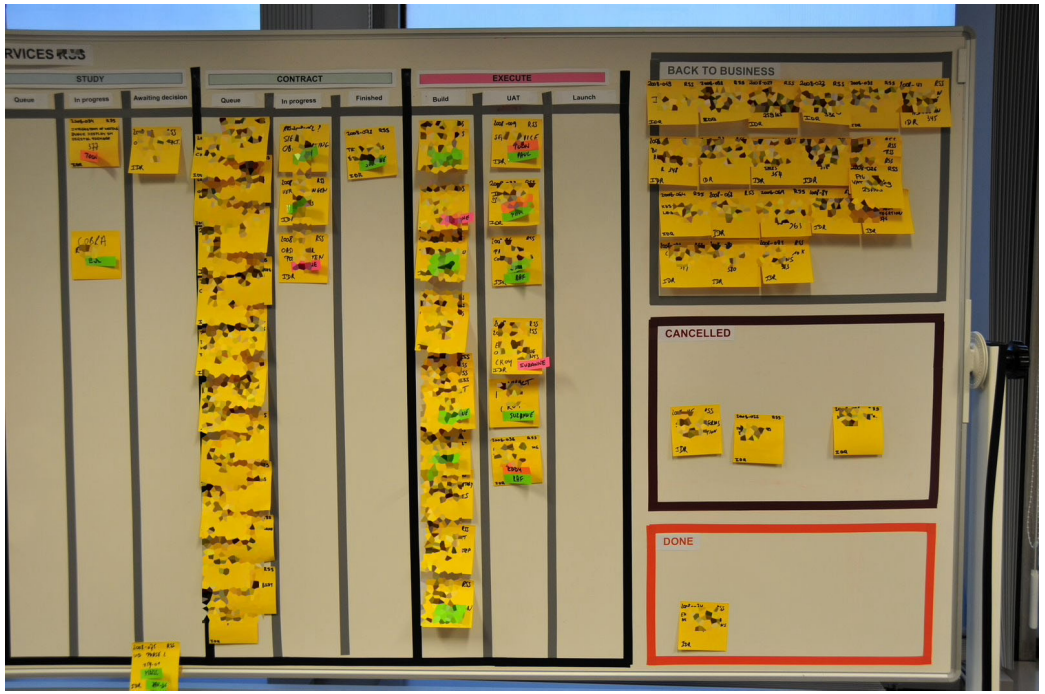
Why we still need Use Cases?

- Problem of User Story:
  - Lack of context
  - Sense of completeness that you covered all bases relating to a goal.
  - No mechanism for looking ahead at upcoming work.

Use of User Stories

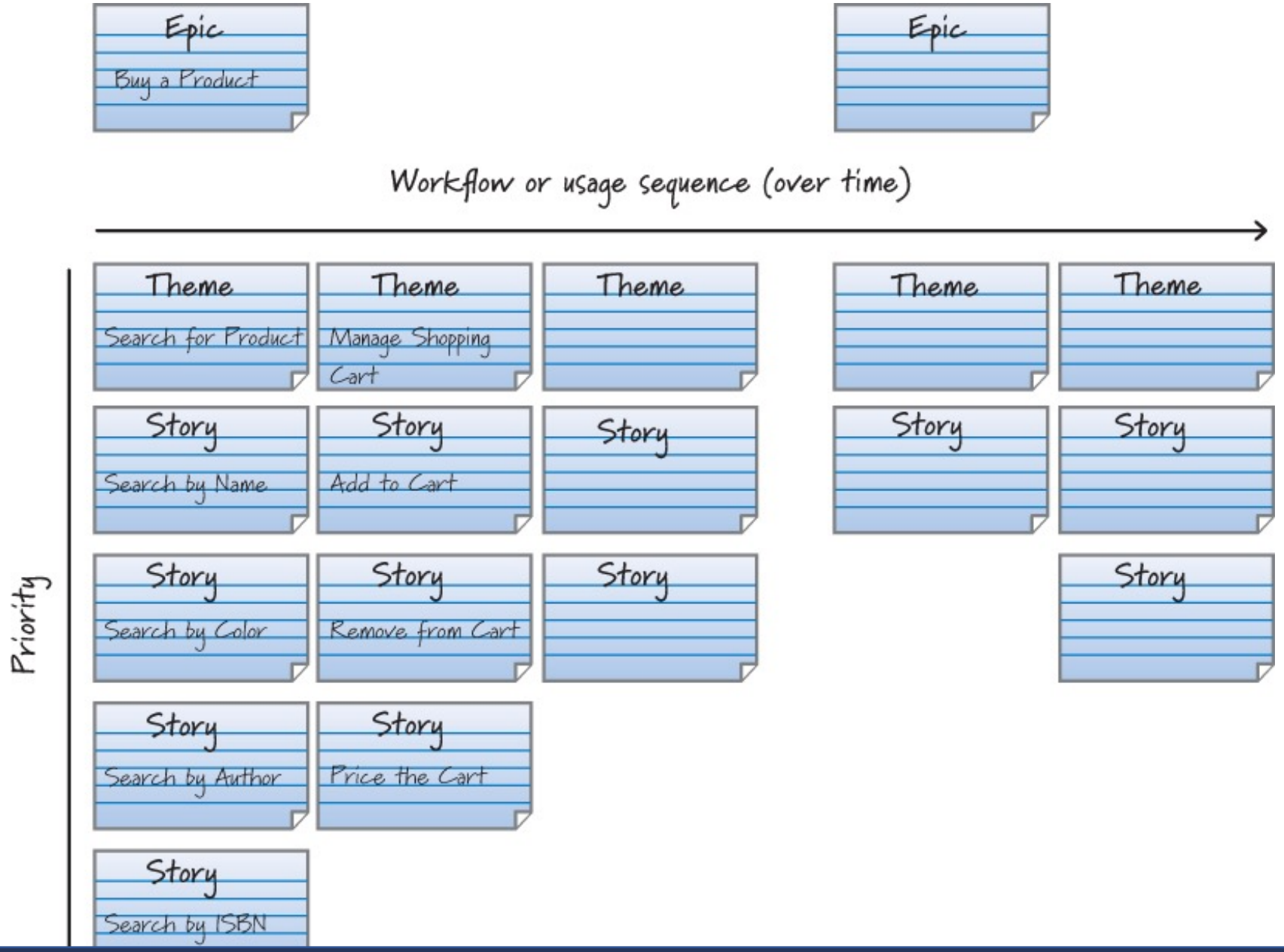
- Keep a board of user stories, group them into “epics”
- **Epics** are large bodies of work that can be broken down into a number of smaller tasks (called stories).



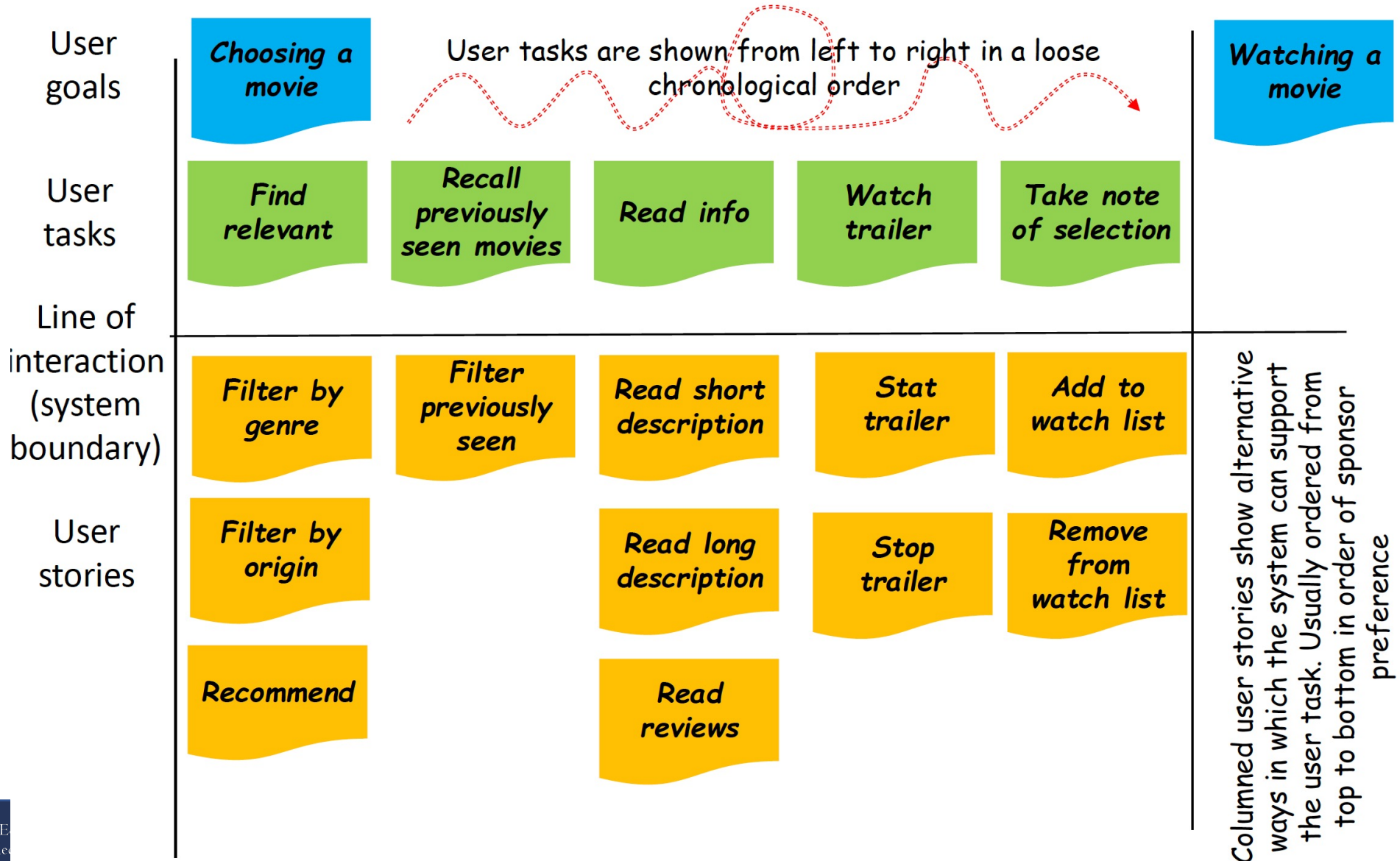


Story Mapping

- Epic
  - Theme
    - Story



Story map for choosing a movie

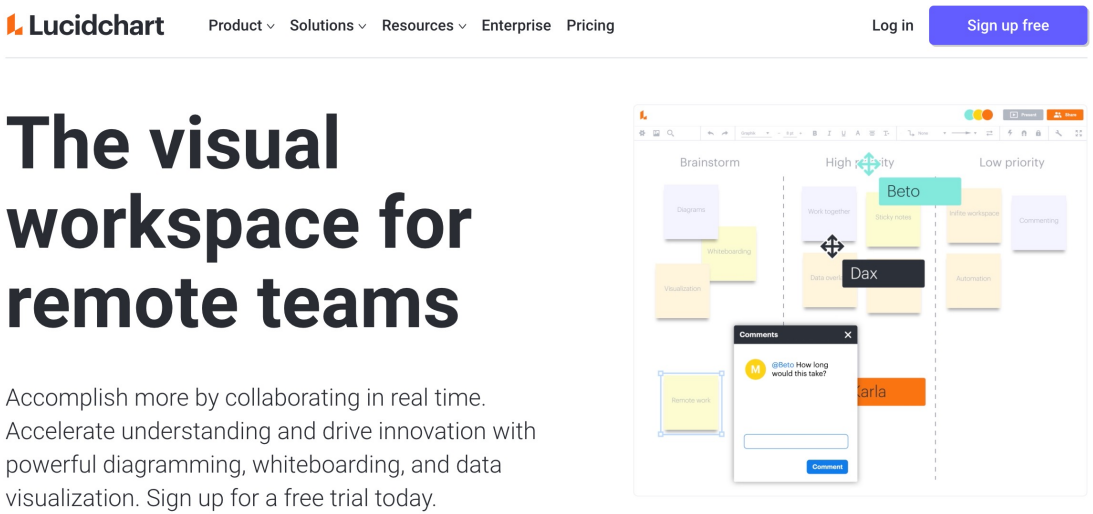


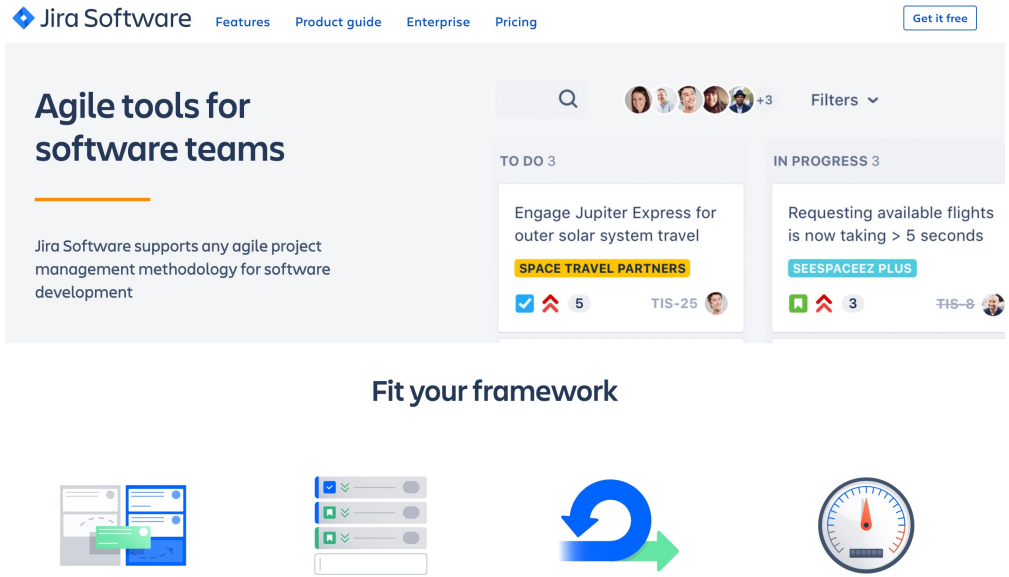
From goals to story maps

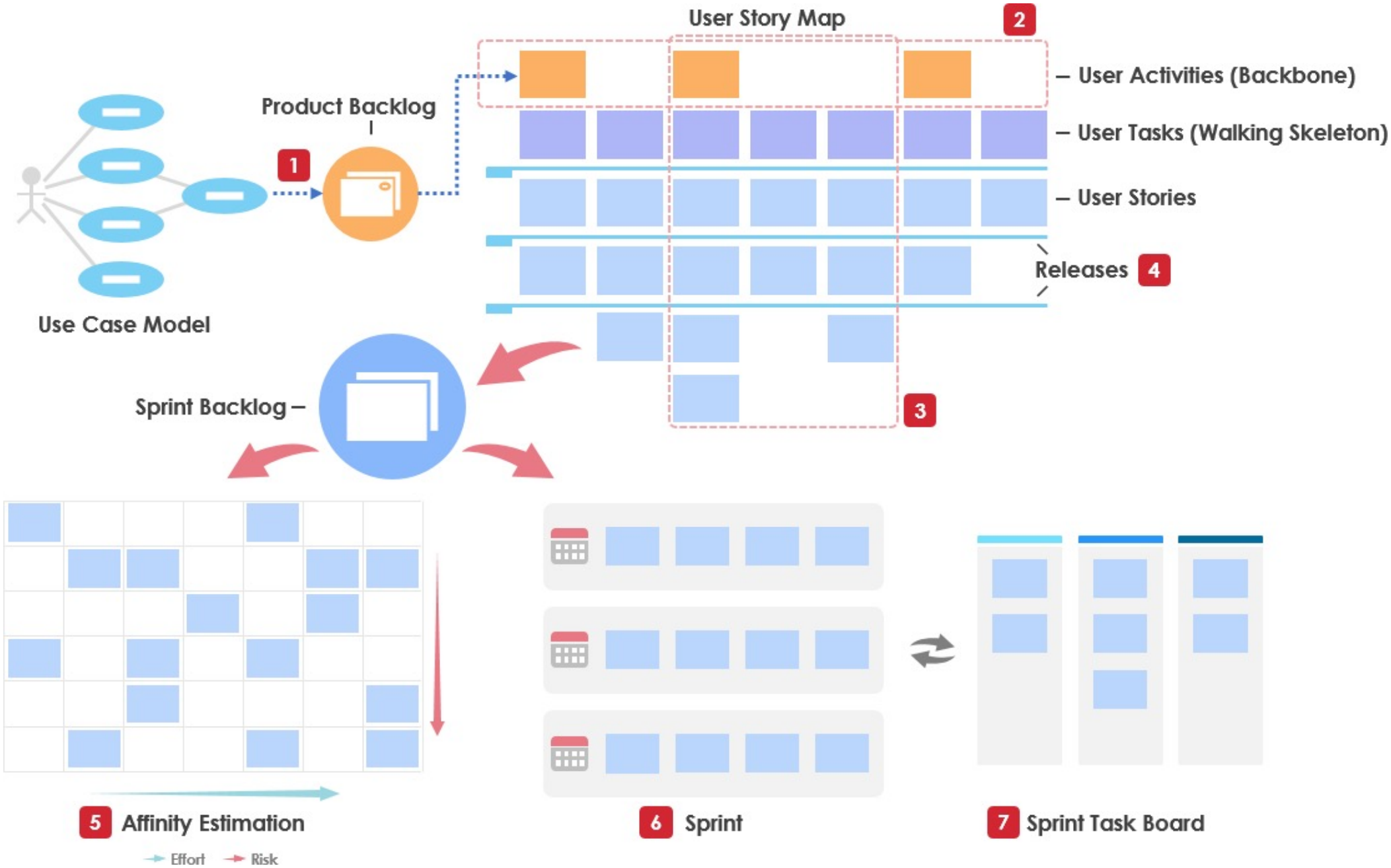
1. Consolidate goals across scenarios (there could be more than one scenario for every given goal)
2. Create a story map column for each consolidated goal
3. Enumerate the tasks you find in the scenarios in the most likely chronological order, do not concern with an strict order, shortcuts, repetitions, etc. Eliminate duplicates
4. Are there missing tasks?
    1. Are there tasks that should precede or succeed any one of the ones you have already included listed?
    2. Are there important task variations that should be considered?
5. What support will the system provide to the user tasks above? List the user stories under the corresponding task in order of preference
    1. Are there alternative ways to support the task?
    2. Does the solution require that the user perform some additional task?
    3. Are there user stories that should precede or succeed the current one?

Integrate Use Case, User Story and Story Mapping techniques

- Lucidchart
- Jira Agile
- Team Foundation Server
- BoardThing
- Stories on Board
- FeatureMap






User Story Map
2
– User Activities (Backbone)
– User Tasks (Walking Skeleton)
– User Stories
Releases
4
Product Backlog
1
Use Case Model
Sprint Backlog –
3
5
Affinity Estimation
Effort
Risk
6
Sprint
7
Sprint Task Board

Industrial Requirements Tools

'Stakeholder Requirements' current 2.2 (Review Phase 3) in /New Family Car Project/Requirements (Formal module) - DO...

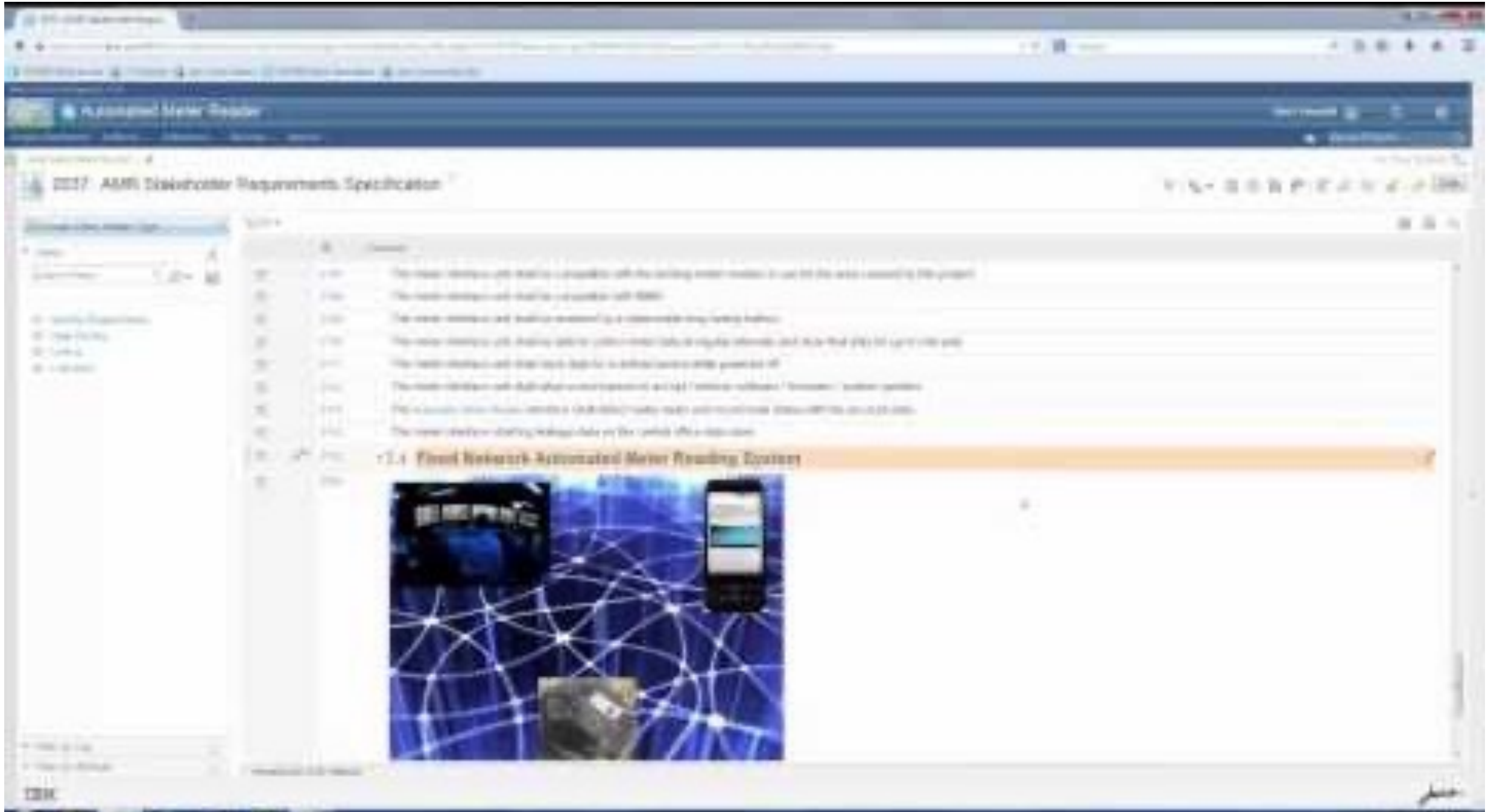
File Edit View Insert Link Analysis Table Tools Discussions User RQM Help

View: 01 - Collect reqts | All levels

- Stakeholder Requirements
  - 1 Introduction
  - 2 User types
    - 2.1 Nationalities
      - The car will be usec
    - 2.2 User sizes
      - People come in all s
  - 3 Requirements
    - 3.1 Capability Requirem
      - 3.1.1 Carrying Capa
      - 3.1.2 Cost Points
      - 3.1.3 Movement
      - 3.1.4 Fuel economy
      - 3.1.5 Safety
      - 3.1.6 Noise levels
      - 3.1.7 Ease of Acces
      - 3.1.8 Visibility
      - 3.1.9 Equipment ma
      - 3.1.10 Entertainmer
      - 3.1.11 Maintenance
      - 3.1.12 Servicing
      - 3.1.13 Indication rec
      - 3.1.14 Terrain
      - 3.1.15 Refueling
    - 3.2 Constraint Requirem
      - 3.2.1 Availability
      - 3.2.2 Lifetime
      - 3.2.3 Security
      - 3.2.4 Accessories

Reqt	Car user requirements	Priority	Acceptability	Queries
TRN-CSR-83	Users shall be able to receive a warning when a service is due.	Mandatory	Acceptable	(JC) What color indicators are we using for the warning system? Are we going to put requirements in place to accomidate this system?
TRN-CSR-84	**3.1.13 Indication requirements**	N/A		
TRN-CSR-85	The user shall be able to see at all times an indication of speed to within + or - 1%.	1	Acceptable	
TRN-CSR-86	The user shall be able to see at all times an indication of engine revolutions to within + or - 1%.	2	Acceptable	
TRN-CSR-92	The user shall be able to obtain direction to go information. 	2	Acceptable	

Username: lbus | Exclusive edit mode



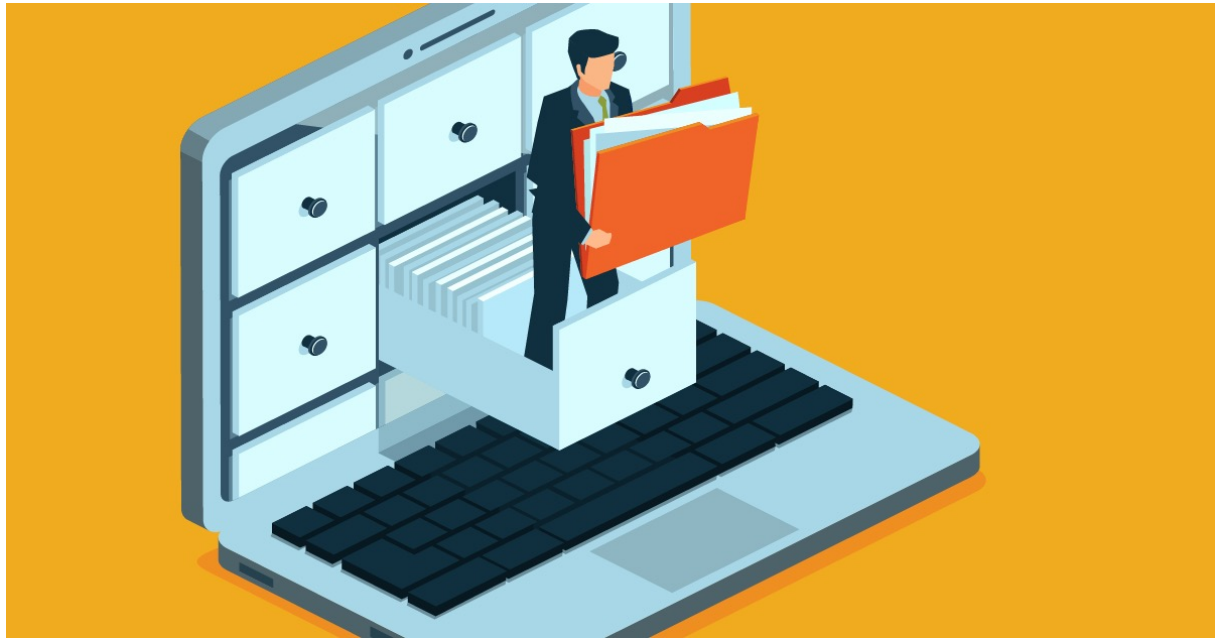
https://www.guru99.com/requirement-management-tools.html

Requirements prioritization

- Cost, time, and other limits
- Dependencies among requirements
- Nice to have
- Strategies to base on value contribution

Product Requirement Document (PRD)

1. Goals
2. User Personas
3. User Stories
4. Functional Requirements
5. Non-Functional Requirements
6. User interaction and design
7. Questions
8. Out of Scope



Summary

- Many documentation strategies; our focus is on *user stories*


RISK

Risk


Tony Webster
@webster
Follow
I appreciate the honesty.
Pick a password
Don't reuse your bank password, we didn't spend a lot on security for this app.
At least 6 characters
your password
Continue
8:20 PM - 15 Sep 2018
5,868 Retweets 15,672 Likes
58
5.9K
16K

What are risks?

- A **risk** is an uncertain factor that may result in a loss of satisfaction of a corresponding objective

For example…

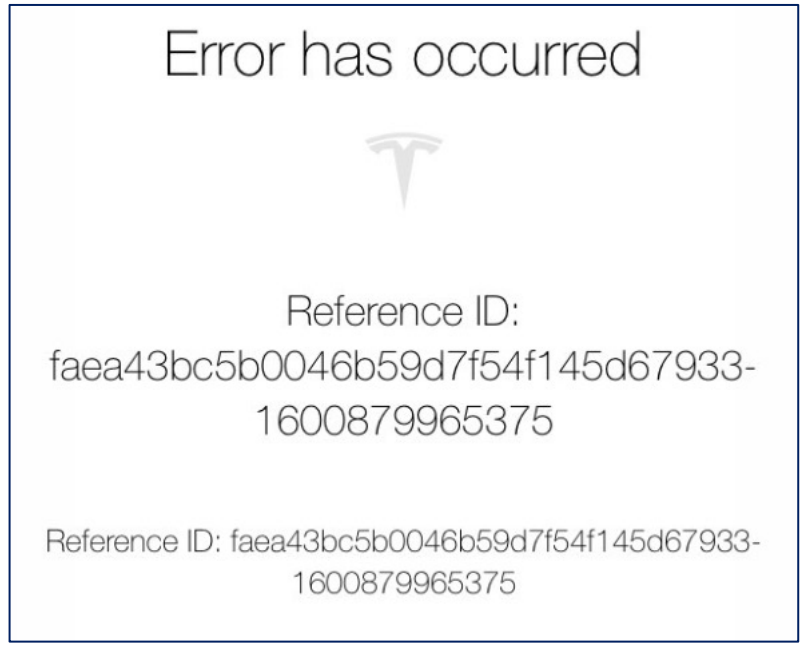
- System delivers a radiation overdose to patients (Therac-25, Theratron-780)
- Medication administration record (MAR) knockout
- Premier Election Solutions vote-dropping “glitch”

Kishore Gopalakrishna @KishoreBytes · Sep 23

I am stranded and cant get into my **Tesla**.

- App won't open
- Keycard is not working
- Emergency roadside assistance- one hour wait time. There is no call me back option.

This is the **error** message. This makes me feel better about the **error** messages in oss projects.



How to assess the level of risk?

- Risks consist of multiple parts:
  - Likelihood of failure
  - Negative consequences or impact of failure
  - Causal agent and weakness (in advanced models)
- Risk = Likelihood x Impact

Aviation failure impact categories

- **No effect** – failure has no impact on safety, aircraft operation, or crew workload
- **Minor** – failure is noticeable, causing passenger inconvenience or flight plan change
- **Major** – failure is significant, causing passenger discomfort and slight workload increase
- **Hazardous** – high workload, serious or fatal injuries
- **Catastrophic** – loss of critical function to safely fly and land

Risk assessment matrix

- MIL-STD-882E

TABLE III. Risk assessment matrix

RISK ASSESSMENT MATRIX				
SEVERITY / PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			

https://www.system-safety.org/Documents/MIL-STD-882E.pdf

Risk Mgmt in Agile

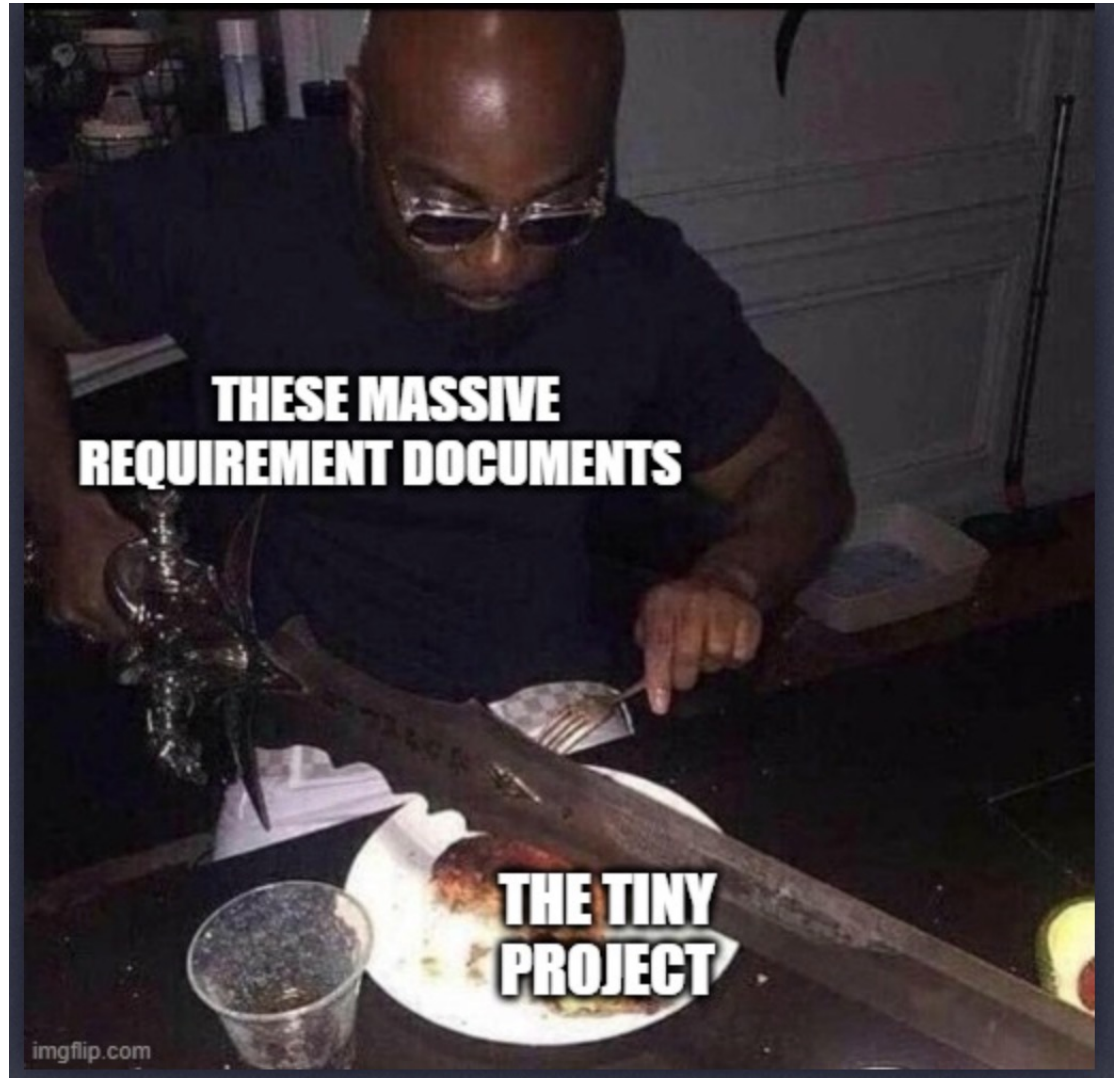
- Short development cycles and quick delivery
- Testing is part of the development cycle
- Business people are often part of the team which reduces risks
- High responsiveness to changes
- Most frameworks do not prescribe risk management processes and techniques which requires the project team to select and adapt adequate measures

Risk Response Strategies

- Accept the risk – for low likelihood or low impact risks, or where cost of mitigation precludes system
- Transfer the risk – push the risk outside the system boundary
- Mitigate the risk – introduce active countermeasures
  - Reduce likelihood of failure
  - Reduce severity of failure
- Avoid the risk – redesign so that risk cannot occur

Risk analysis example (Time Keeper)

	Risk	Probability	Impact	Solution
1	Application crashes	Low	Medium	Introduce Long-term stability test
2	Inappropriate auto-scheduling	Medium	Low	Adjust auto-generated schedule manually
3	Outdated integration	Low	Low	Ignore

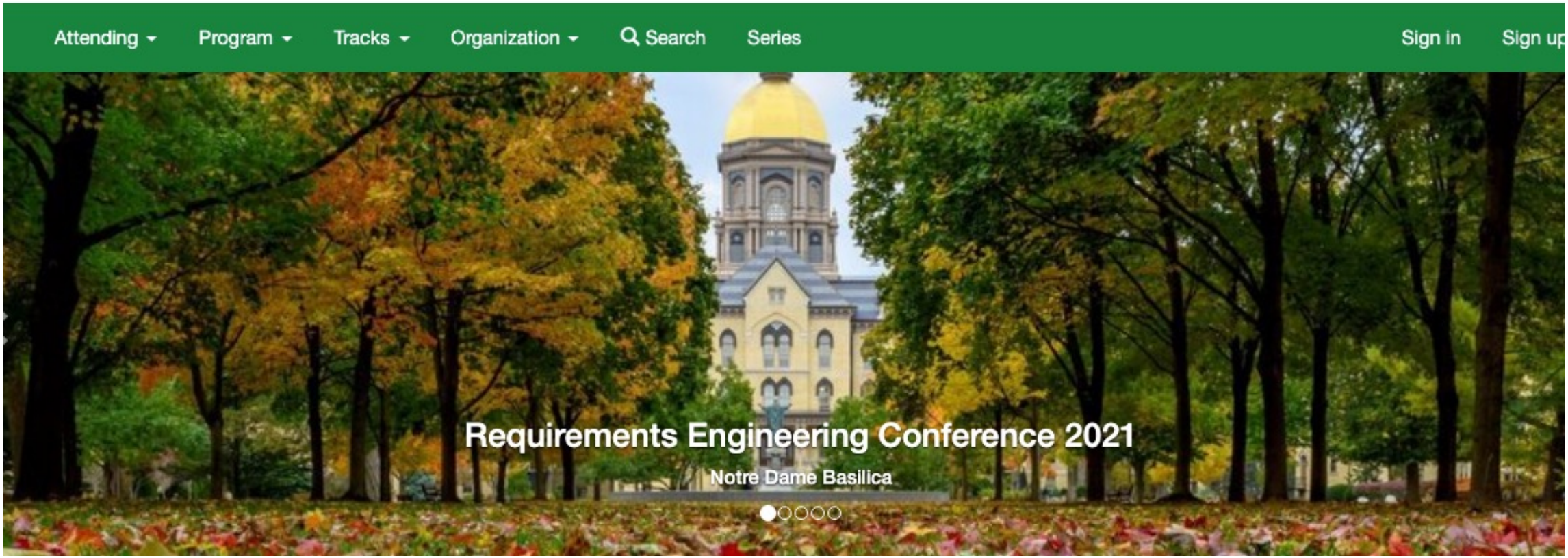

THESE MASSIVE
REQUIREMENT DOCUMENTS
THE TINY
PROJECT
imgflip.com

Further Reading

- Larman, Craig. *Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Interative Development*. Pearson, 2012. Chap. 6
- Van Lamsweerde A. Requirements engineering: From system goals to UML models to software. John Wiley & Sons; 2009. Chapter 2-4
- “Advanced Use Case Modeling, Volume I”, Frank Armour, Granville Miller,Addison-Wesley, 2001, Ch 8-10.
- https://aanimesh.files.wordpress.com/2013/09/applying-uml-and-patterns-3rd.pdf

29th IEEE International Requirements Engineering Conference

Notre Dame, South Bend, USA
September 20-24, 2021



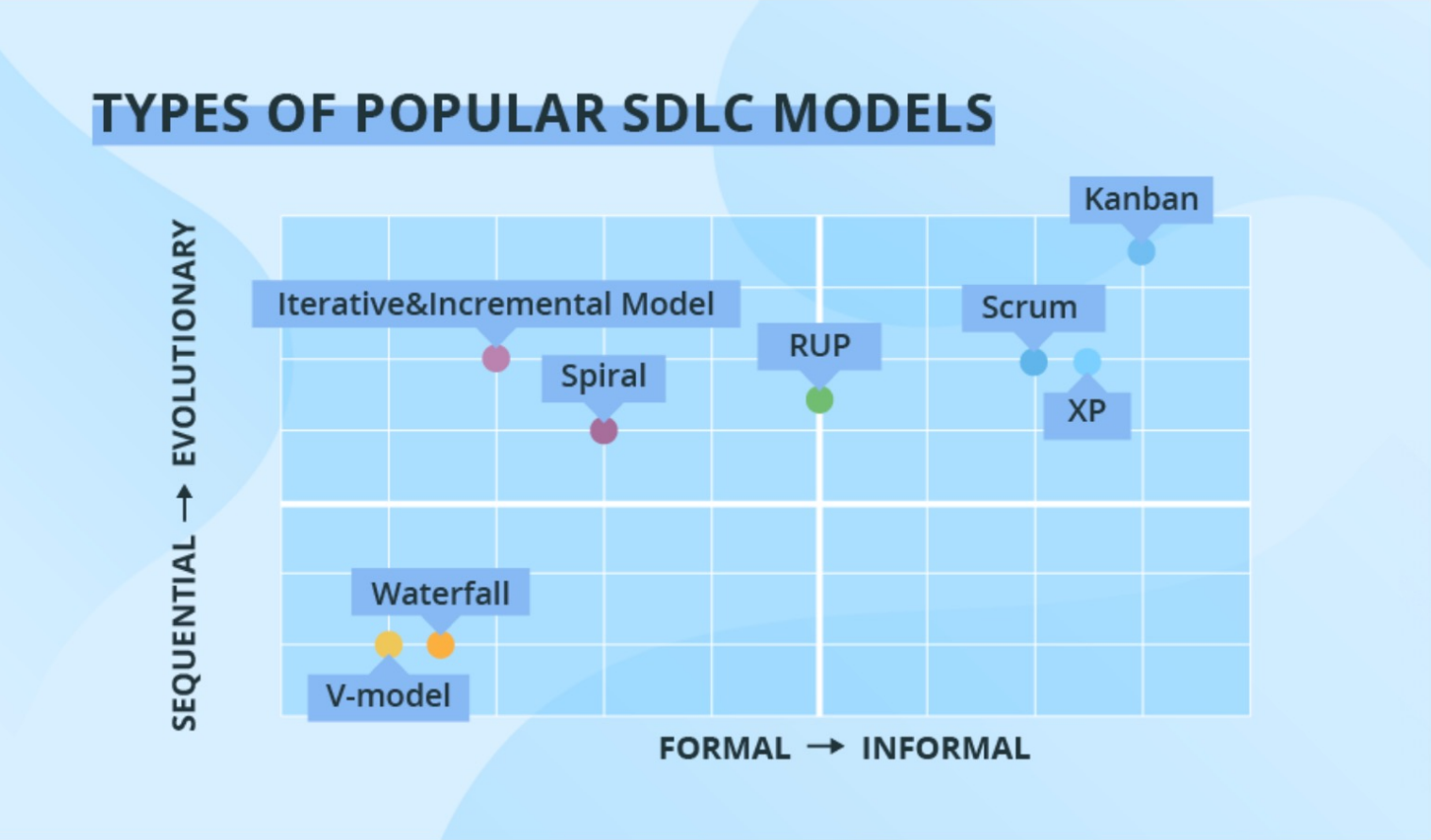
Software Development Models

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Learning Goals

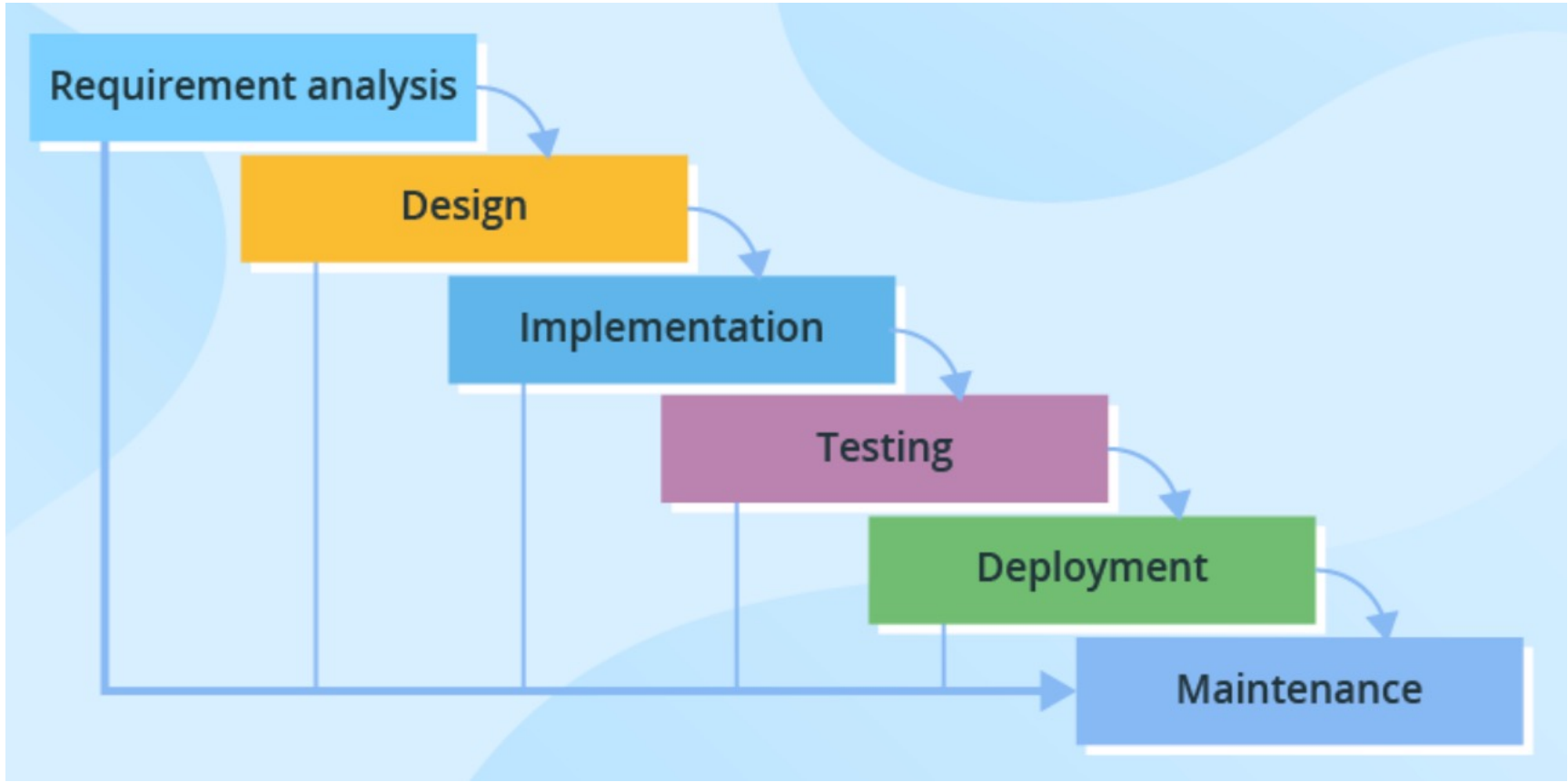
- Understand the concepts and differences between different development model
- Define agile as both a set of iterative process practices and a business approach for aligning customer needs with development.
- Explain the motivation behind and reason about the tradeoffs presented by several common agile practices.
- Summarize both scrum and extreme programming, and provide motivation and tradeoffs behind their practices.
- Identify and justify the process practices from the agile tradition that are most appropriate in a given modern development process.

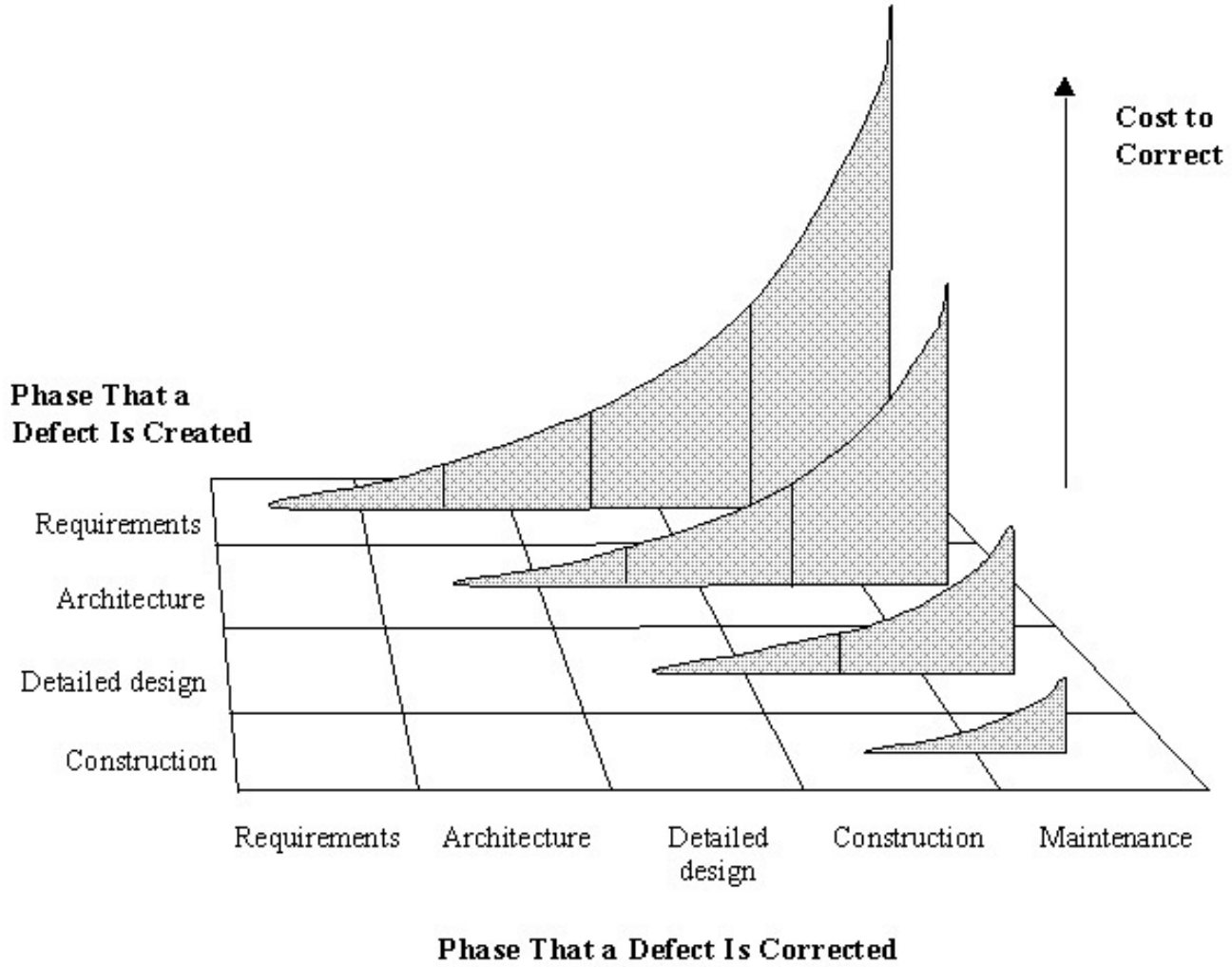
Software Development Models



https://www.scnsoft.com/blog/software-development-models

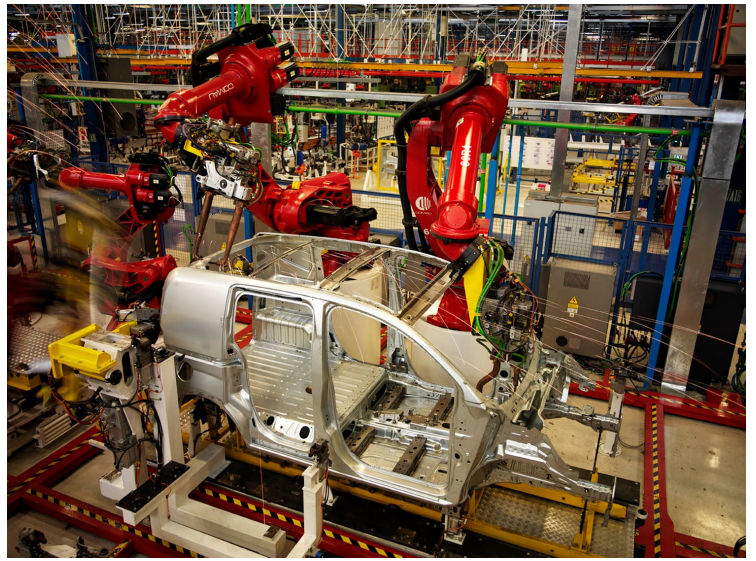
Waterfall Model





Producing a car/bridge

- Estimable costs and risks
- Expected results
- High quality
- Separation between plan and production
- Simulation before construction
- Quality assurance through measurement
- Potential for automation

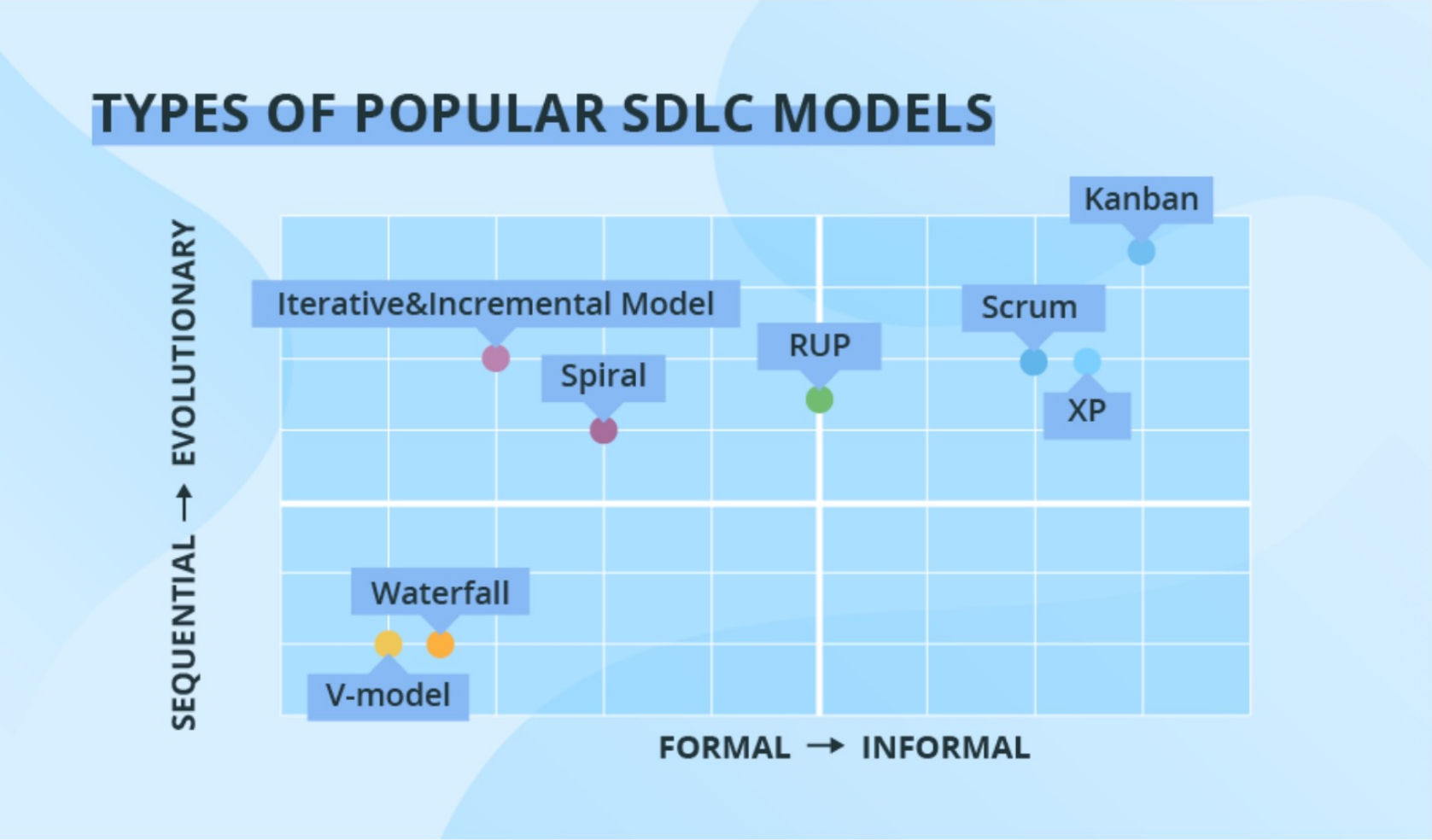


The fact....

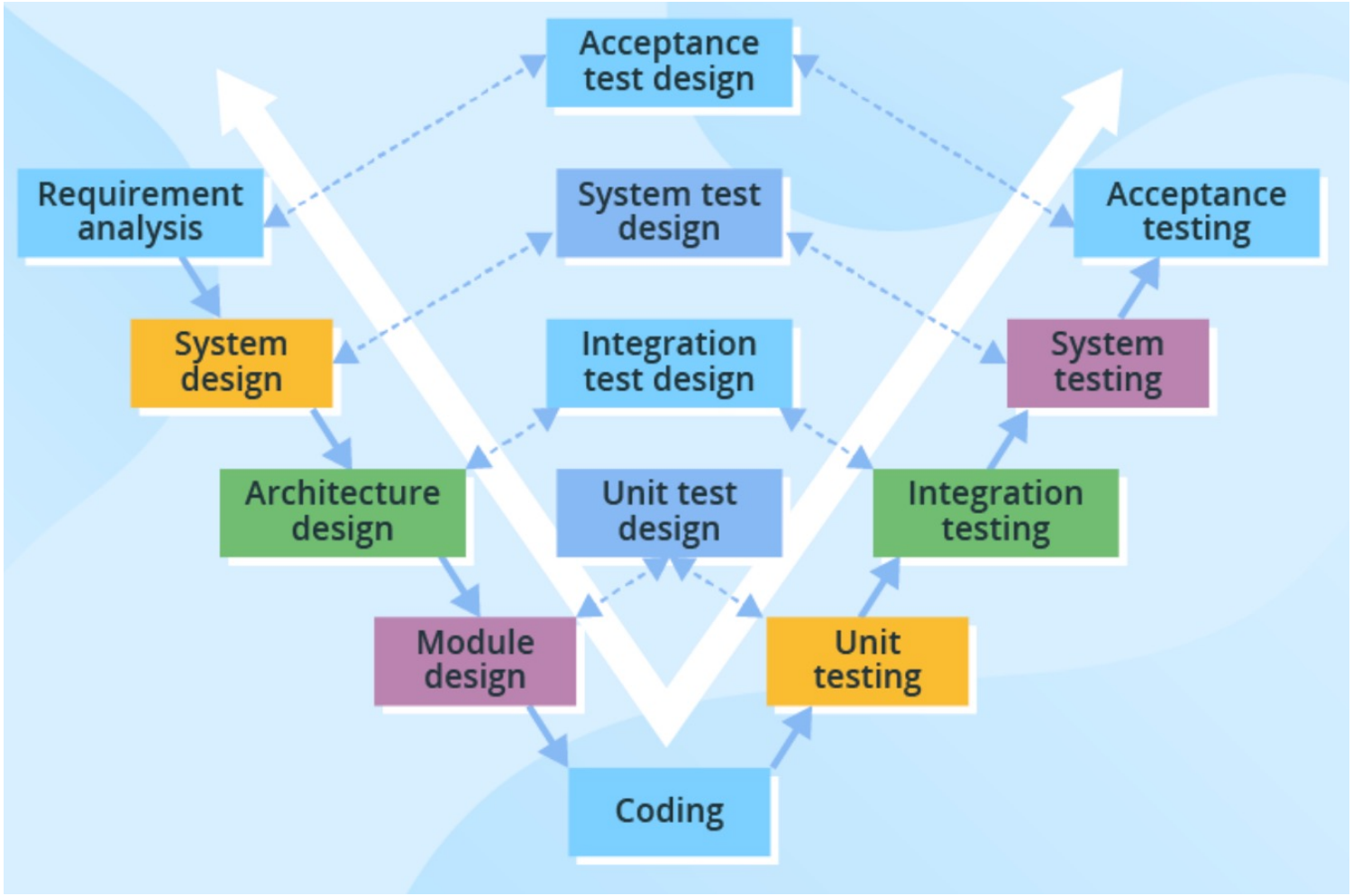
- Properties of software vastly different than for traditional engineering systems
- Software systems are abstract and intangible;
  not constrained by physical laws or manufacturing processes
- No natural limits to the potential of software
  e.g., singularity
- It probably doesn't make sense to apply typical engineering methods and processes to software.

Software Development Models

https://www.scnsoft.com/blog/software-development-models



V-Model

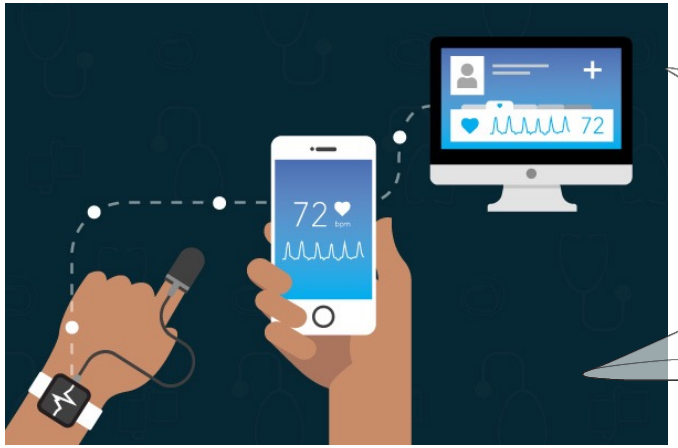


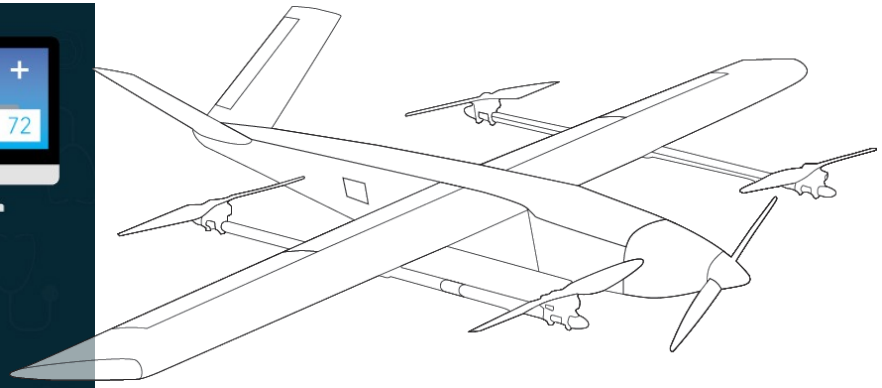
Expensive and time-consuming

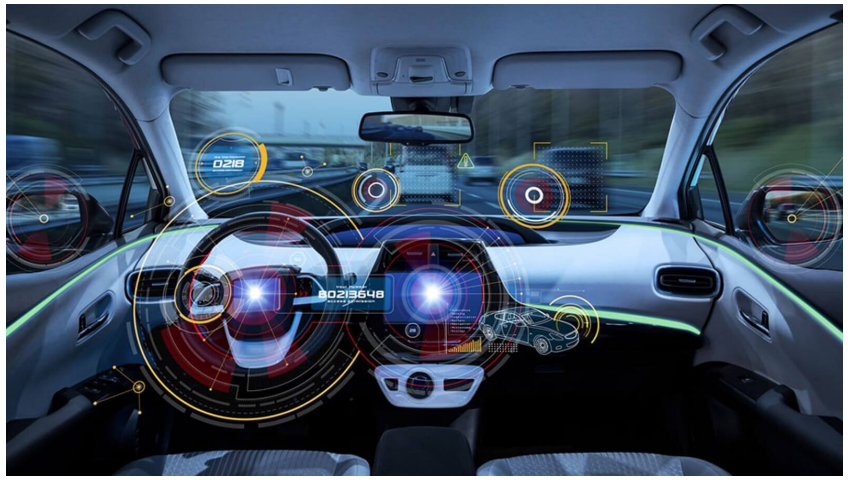
https://www.youtube.com/watch?time_continue=6&v=An7HC1LolDM&feature=emb_logo

Use cases

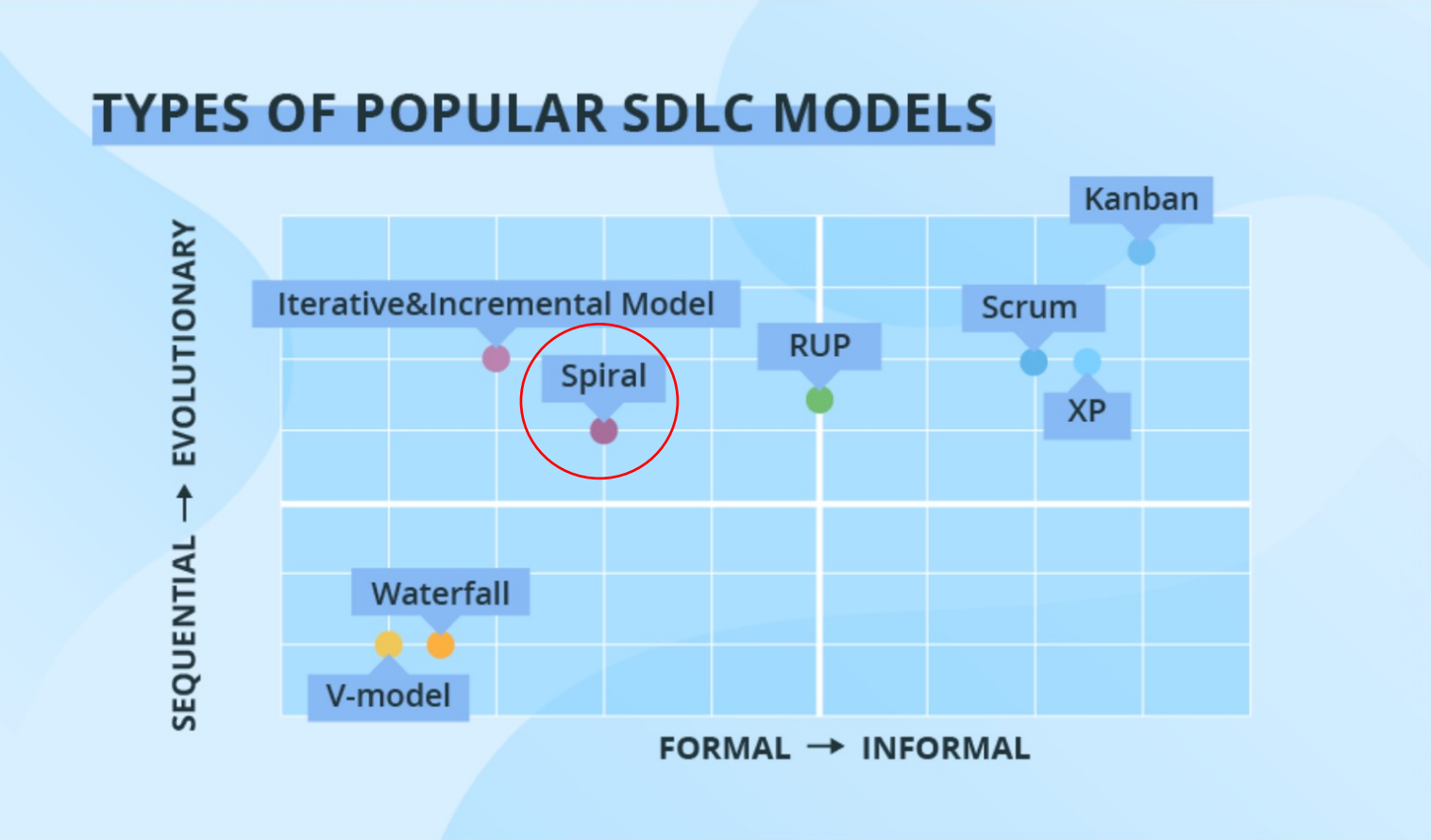
Projects where failures and downtimes are unacceptable (e.g., medical software, aviation fleet management software).







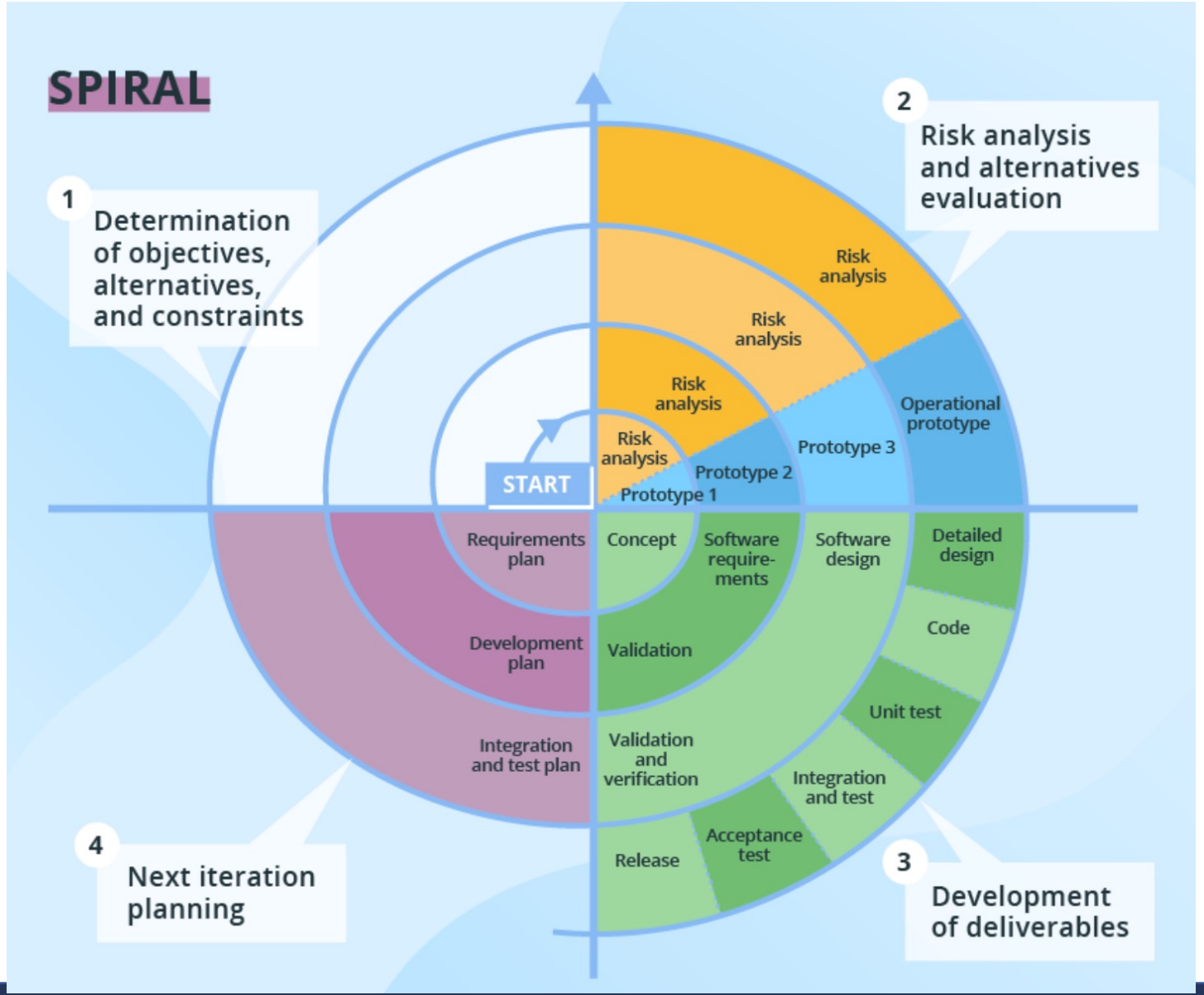
Software Development Models



https://www.scnsoft.com/blog/software-development-models

Spiral

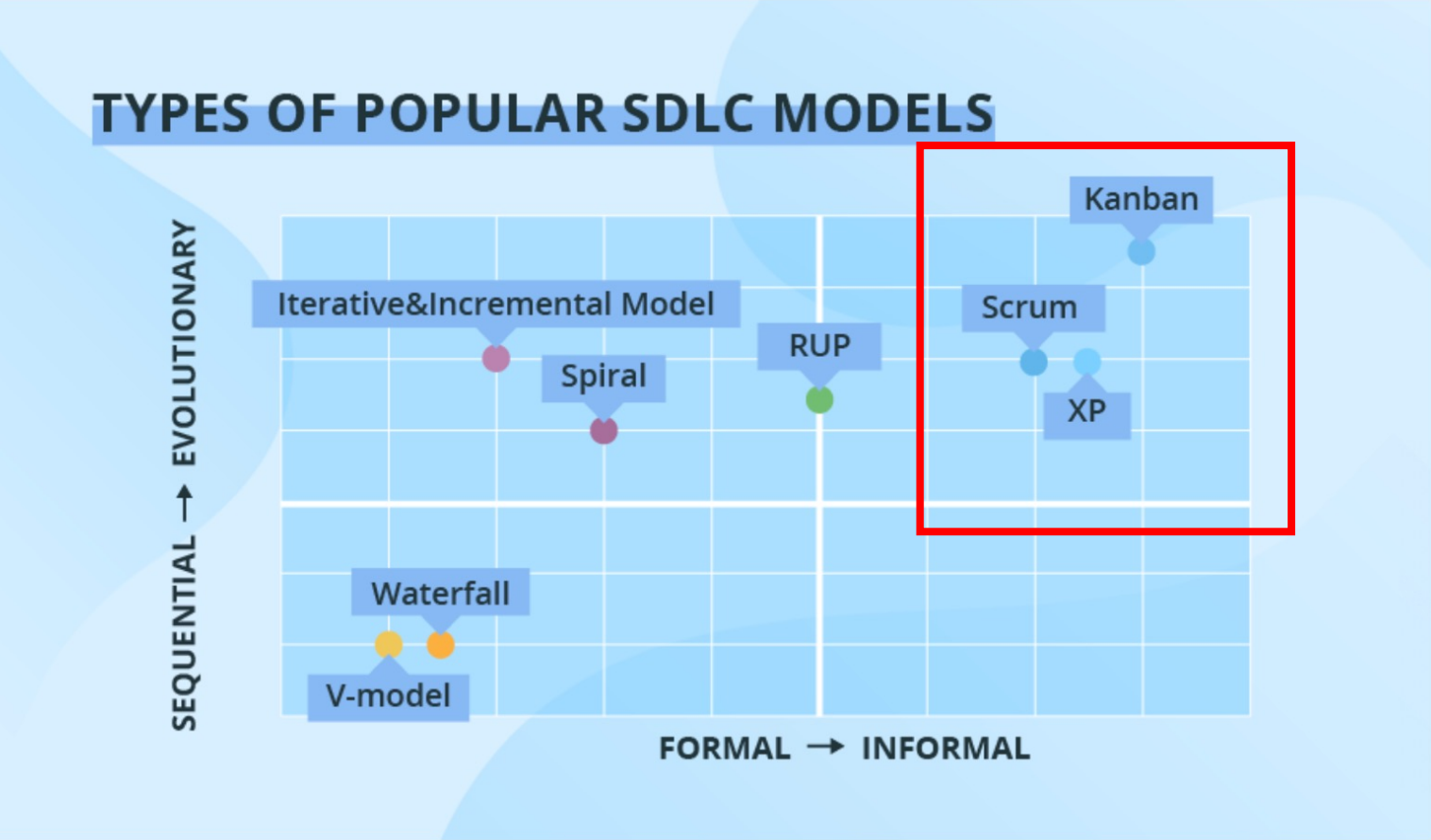
- Focus on thorough risk assessment.
- Intensive customer involvement



Use Cases

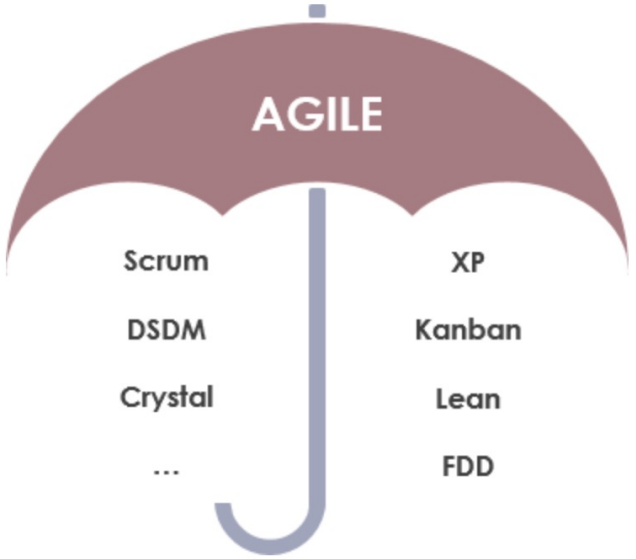
- Projects with unclear business needs or too ambitious/innovative requirements.
- Projects that are large and complicated.
- Research and development (R&D) activity or the introduction of a new service or a product.

Software Development Models

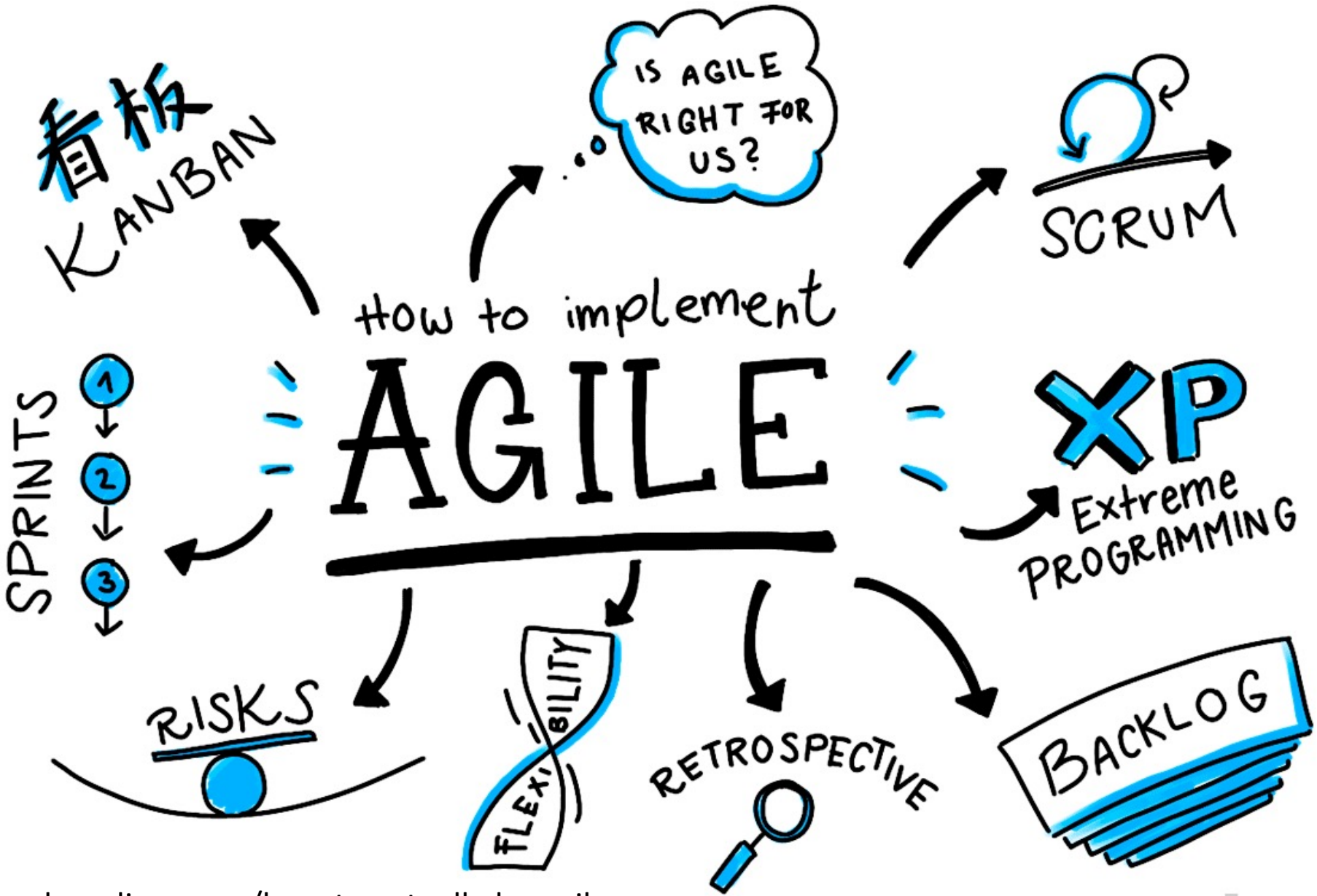


https://www.scnsoft.com/blog/software-development-models

Agile


AGILE
Scrum
DSDM
Crystal
...
XP
Kanban
Lean
FDD

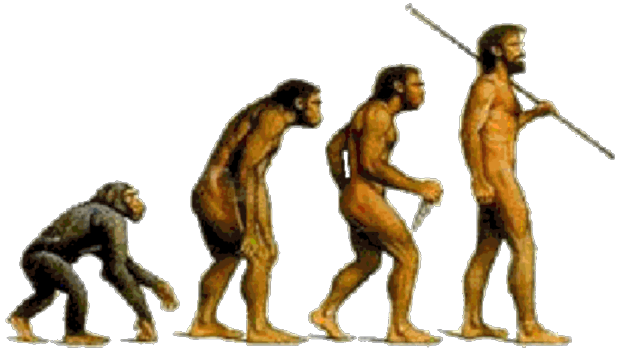

WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.
www.dilbert.com scottadams@aol.com
© Scott Adams, Inc./Dist. by UFS, Inc.



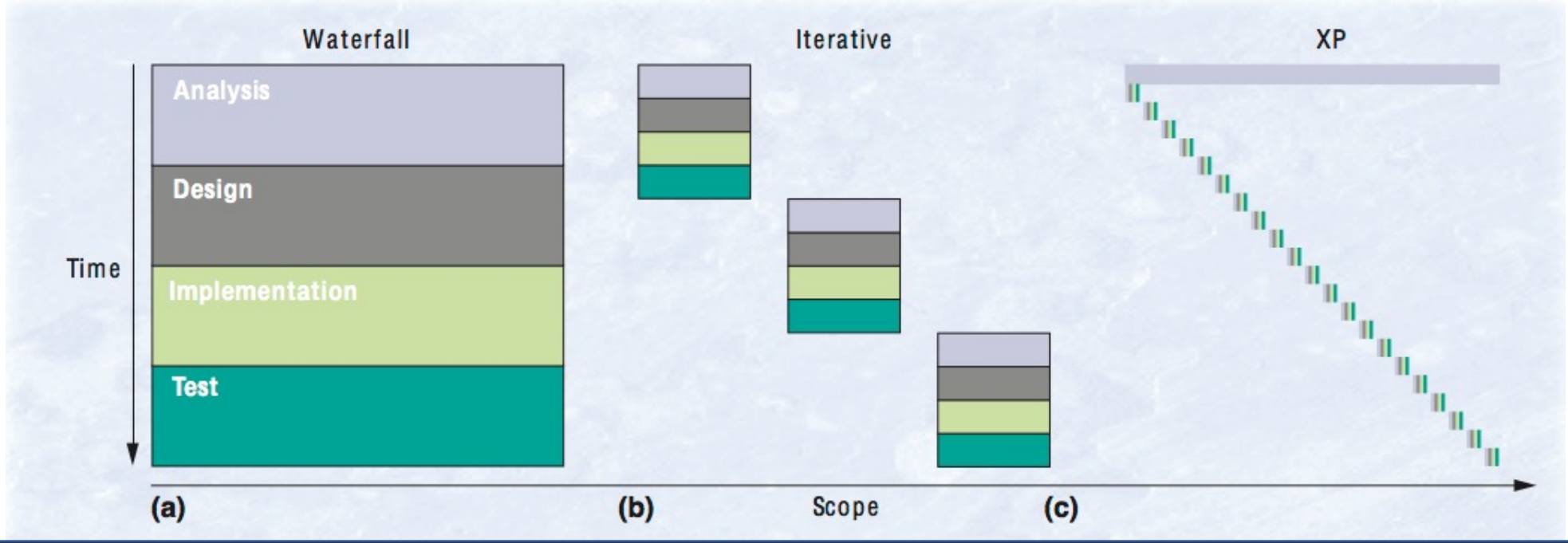
https://sredmond.medium.com/how-to-actually-be-agile-f5f15c6b2de7

Extreme Programming (XP)

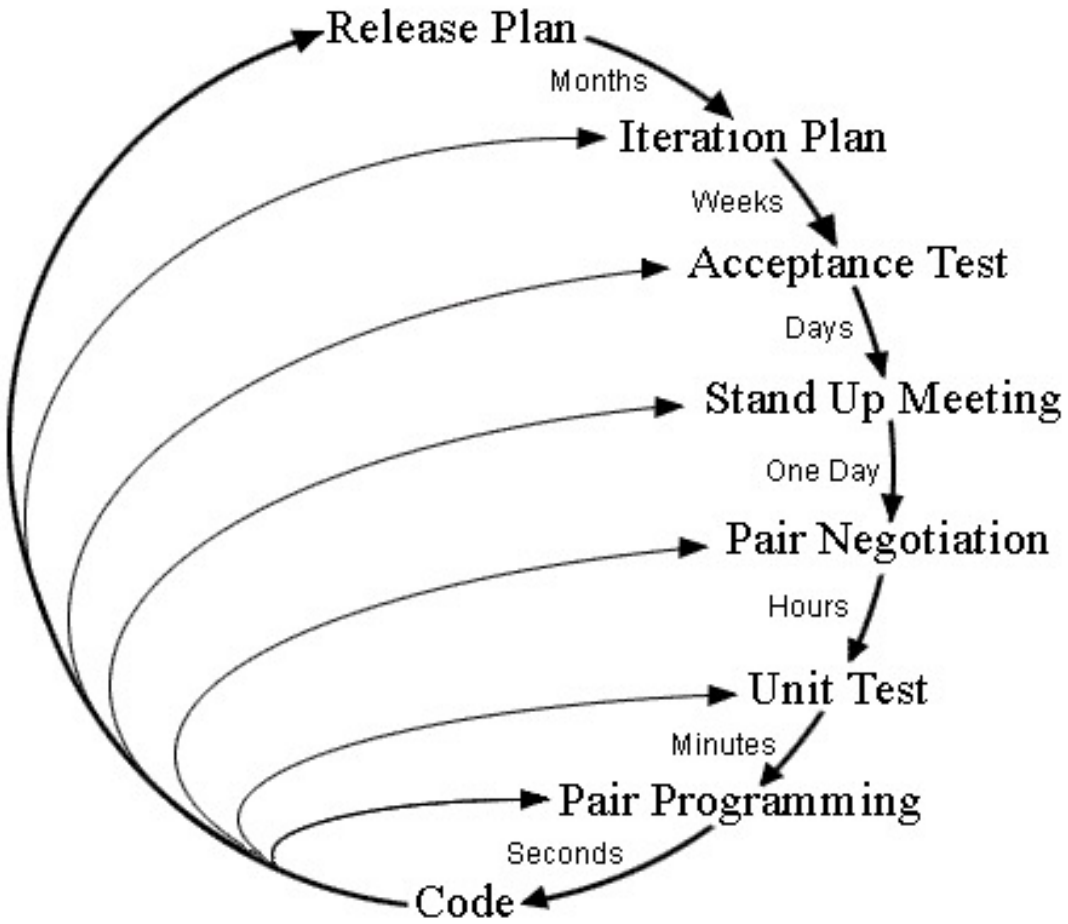
Human evolution



XP evolution



Extreme Programming (XP)

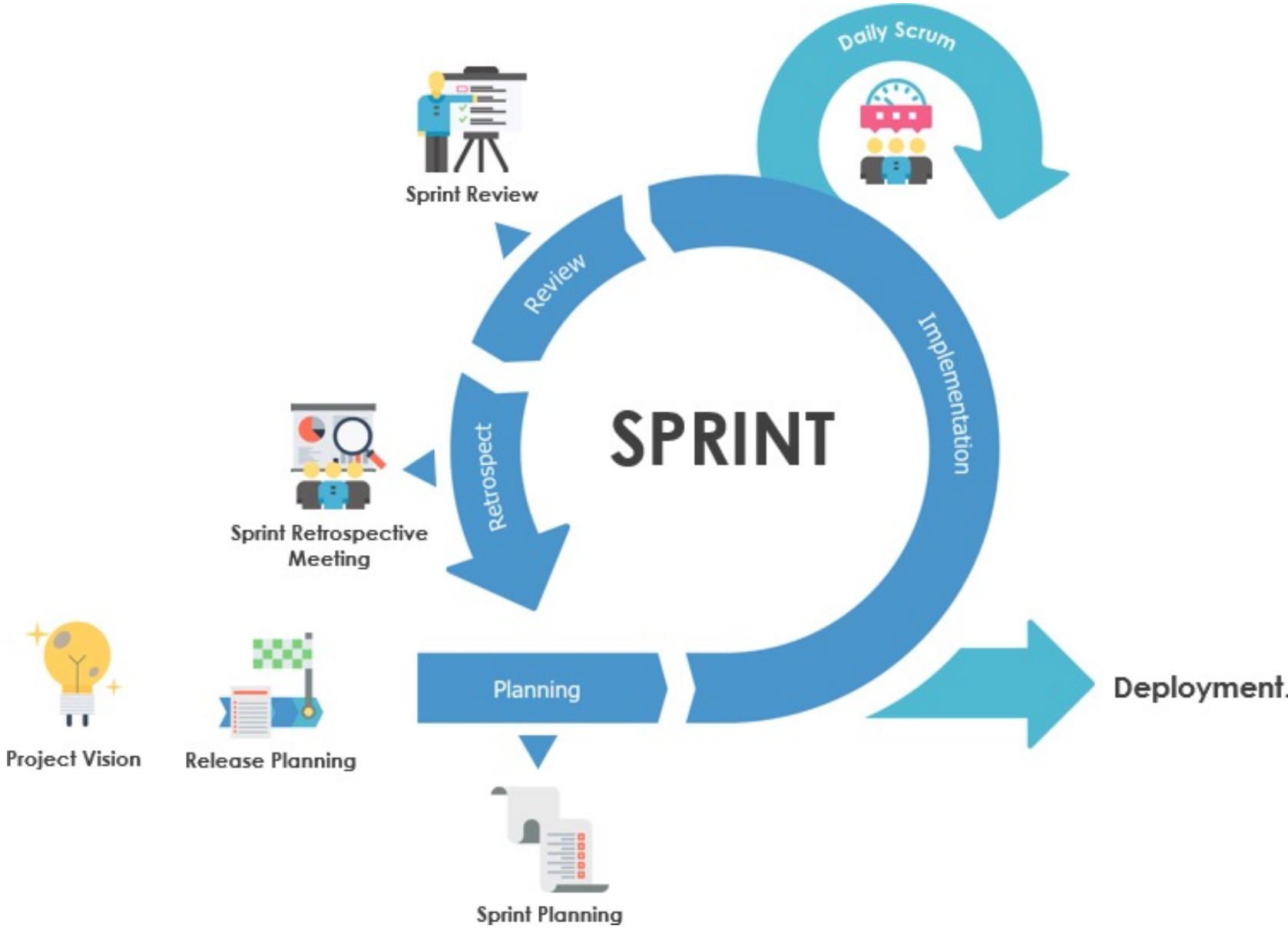


XP Practices (subset of Agile!)

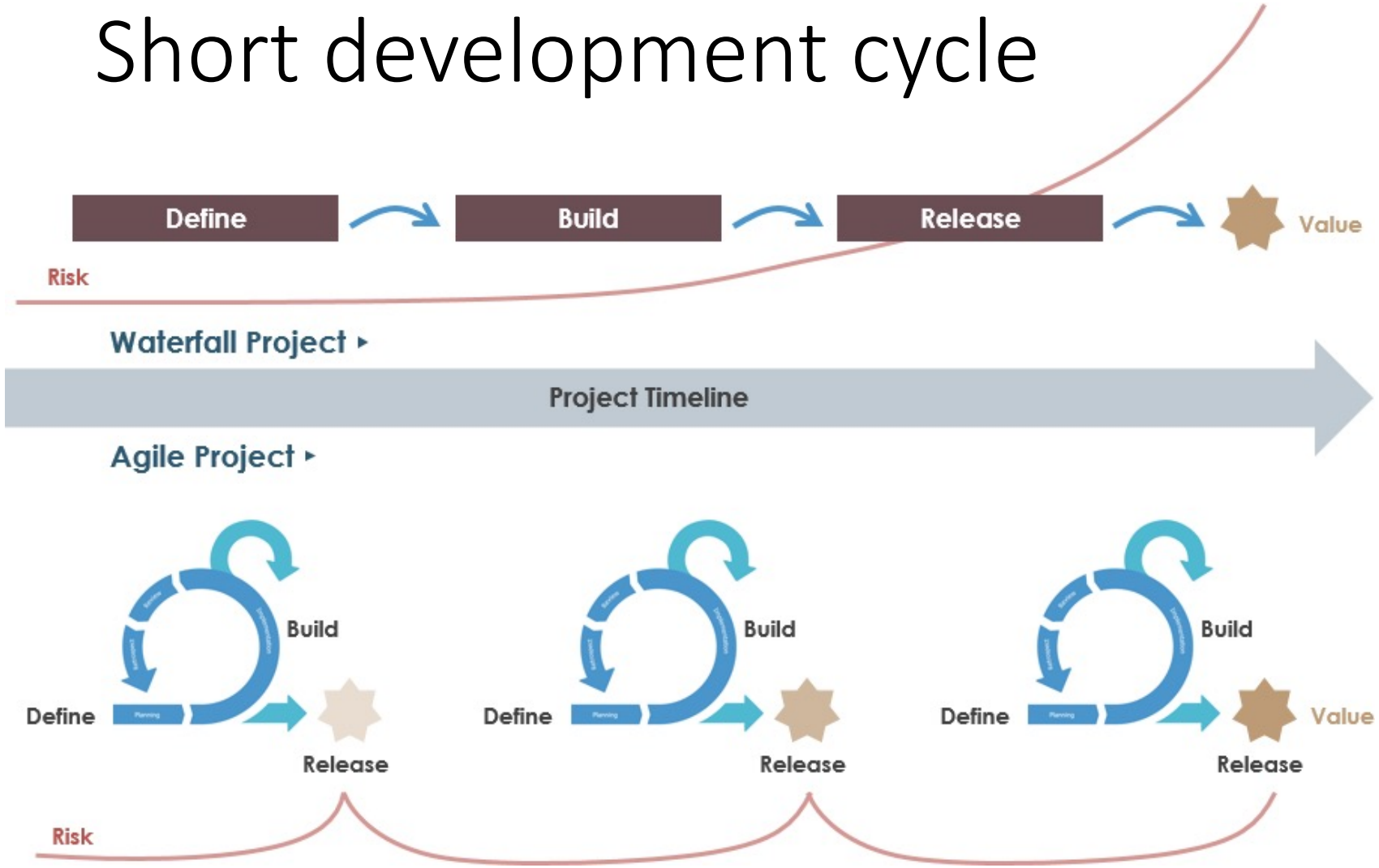
- TDD (test-first approach).
- Planning game: 1-3 week iterations, one iteration at a time, customer decides which user stories to use
- Whole team/on-site customer: “customer speaks with one voice.” Customer may be a whole team.
- Small releases, with valuable functionality, to guard against unhappy customers.
- System metaphor is a single shared story of how it works. (Sort of like architecture)
- Simplest thing that possibly works (coding for today)
- Refactor all the time, because you don’t have up-front design before programming.
- Collective ownership. Everyone is responsible for everything. If a programmer sees something she doesn’t like, she can go change it. Task ownership is individual.
- Pair programming. can code alone for nonproduction code like prototypes
- Continuous Integration. A day of development at most.
- Sustainable pace. 40 hour work weeks.
- Coding standards, Especially since all code can change at all times.

Scrum Process

- Sprint Cycle



Short development cycle



Agile Software Development Is …

Both:

- a set of software engineering best practices (allowing for rapid delivery of high quality software)
- a business approach (aligning development with customer needs and goals)

On-site Customer

A customer sits with the team full-time.

The Manifesto for Agile Software Development (2001)

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

The Twelve Principles of Agile Software Development

Individuals and interactions

1. Projects are built around motivated individuals, who should be trusted
2. Face-to-face conversation is the best form of communication (co-location)
3. Self-organizing teams

Working software

4. Working software is delivered frequently (weeks rather than months)
5. Working software is the principal measure of progress
6. Sustainable development, able to maintain a constant pace
7. Continuous attention to technical excellence and good design
8. Simplicity—the art of maximizing the amount of work not done—is essential

Customer collaboration

9. Customer satisfaction by rapid delivery of useful software
10. Close, daily cooperation between business people and developers

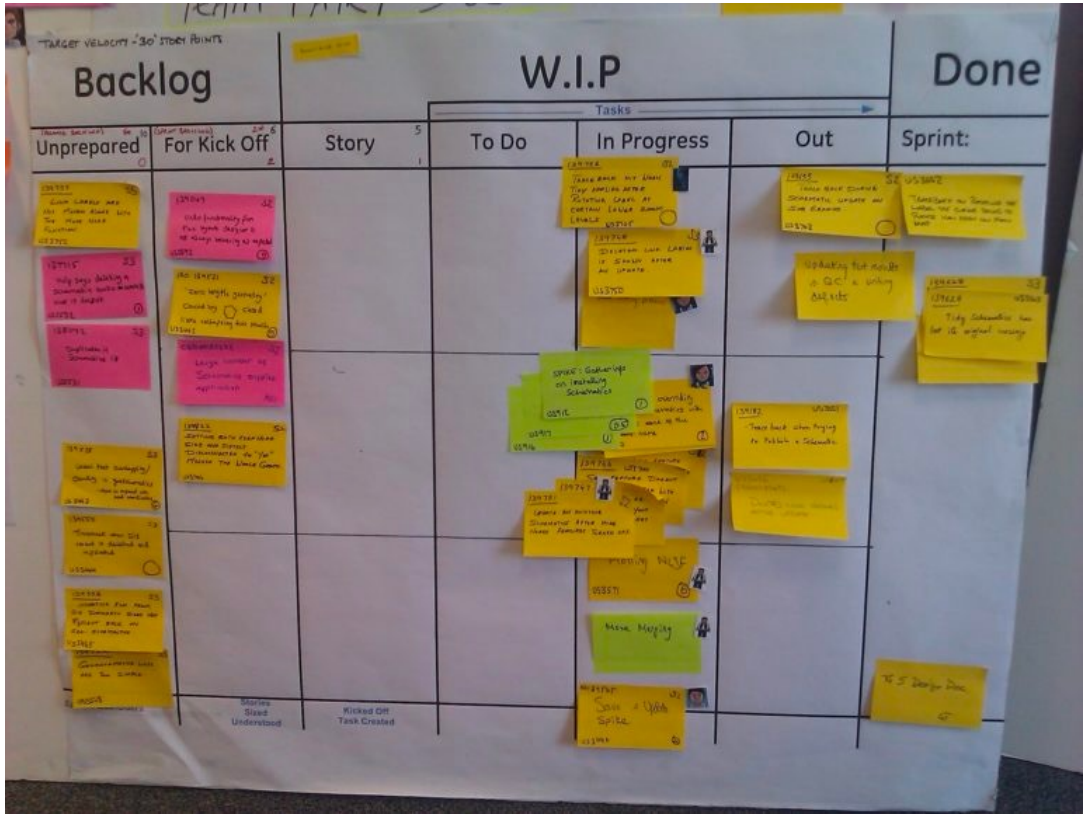
Responding to change

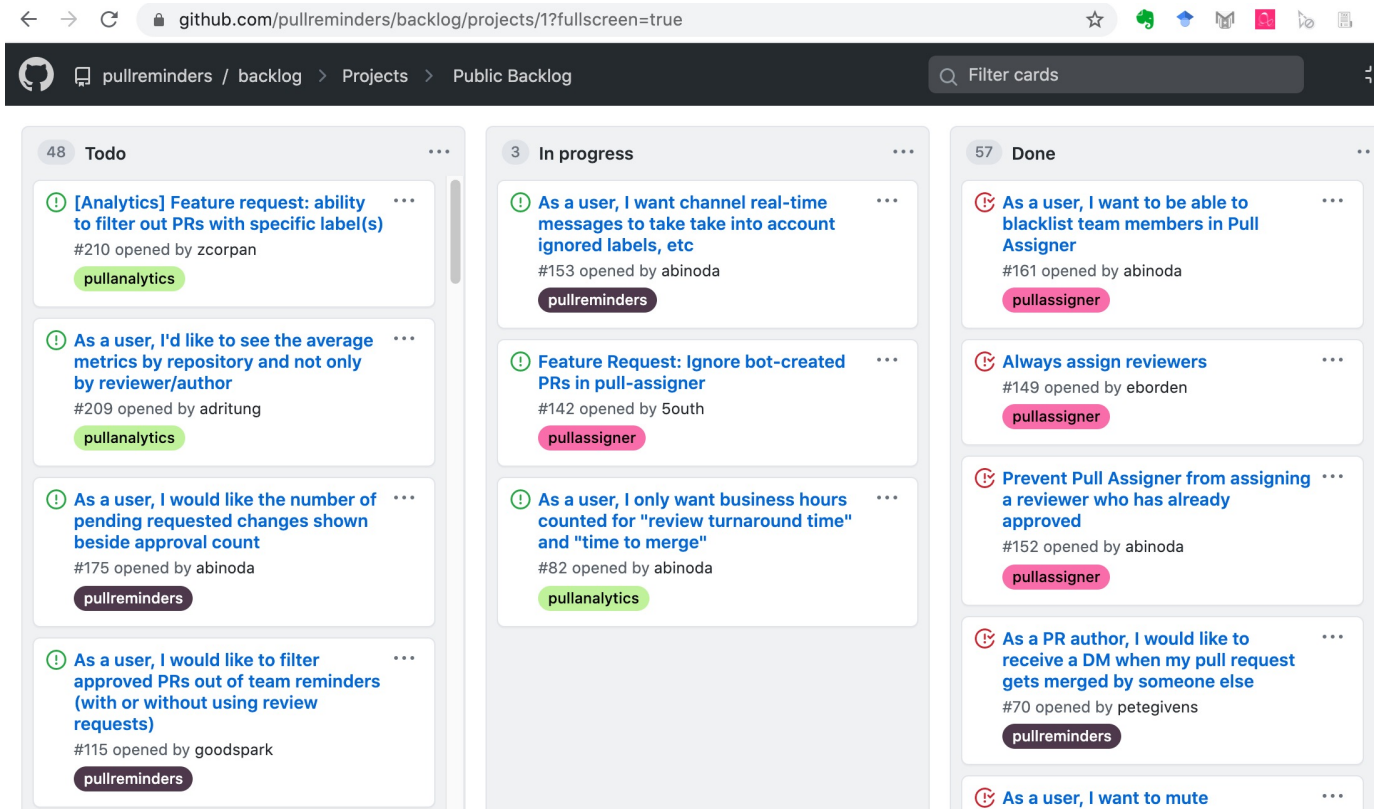
11. Welcome changing requirements, even late in development
12. Regular adaptation to changing circumstances

Agile Practices

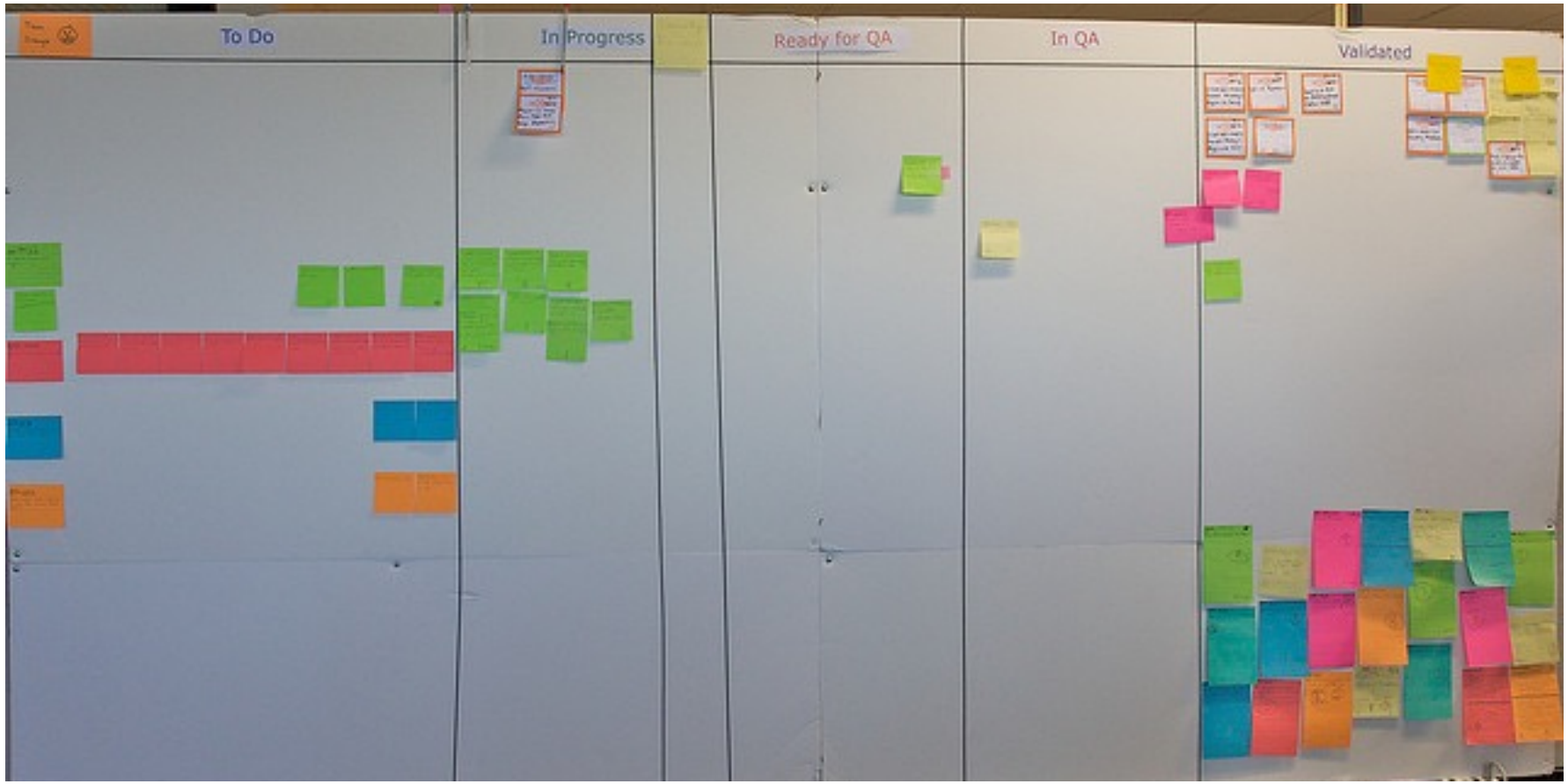
- Backlogs (Product and Sprint)
- Behavior-driven development (BDD)
- Cross-functional team
- Continuous integration (CI)
- Domain-driven design (DDD)
- Information radiators (Kanban board, Task board, Burndown chart)
- Acceptance test-driven development (ATDD)
- Iterative and incremental development (IID)
- Pair programming
- Planning poker
- Refactoring
- Scrum meetings (Sprint planning, Daily scrum, Sprint review and retrospective)
- Small releases
- Simple design
- Test-driven development (TDD)
- Agile testing
- Timeboxing
- Use case
- User story
- Story-driven modeling
- Retrospective
- On-site customer
- Agile Modeling
- 40-hour weeks
- Short development cycles
- Collective ownership
- Open workspace
- Velocity tracking
- Etc.

Backlog – information radiators





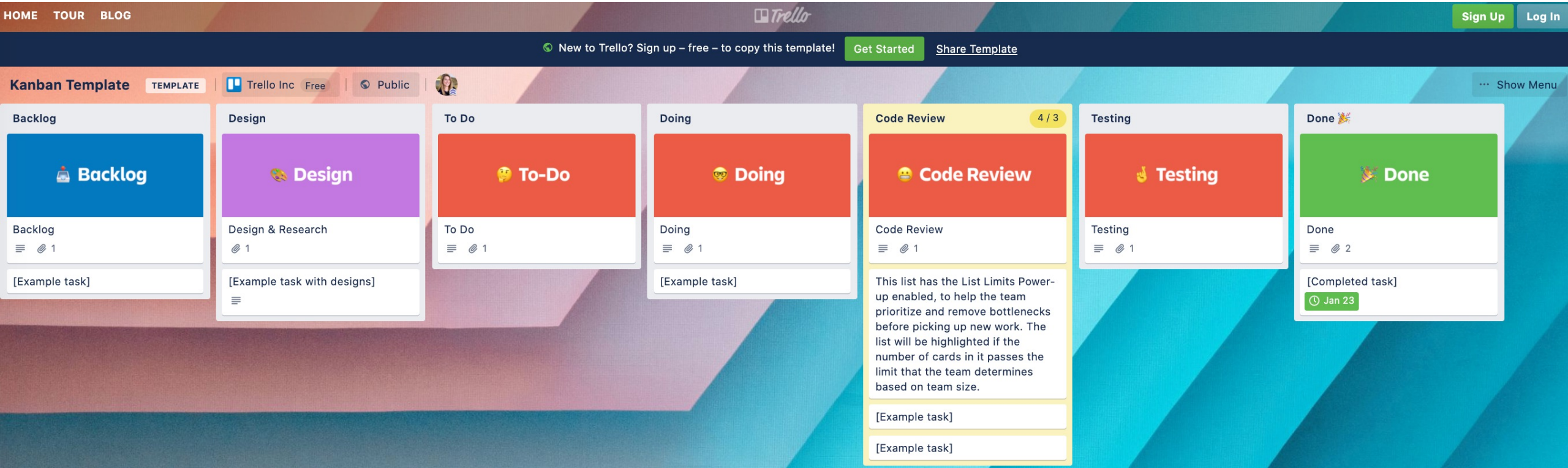
Kanban Board



Kanban board structure

- Visual signals (typically cards)
- Columns or lists
- Work-in-progress limits
- A commitment point
- A delivery point

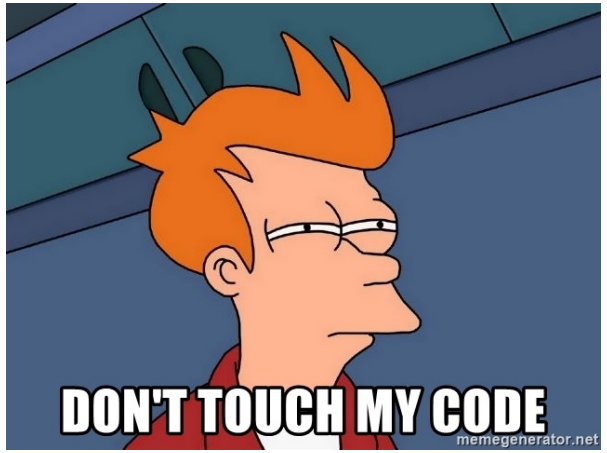
https://trello.com/b/LGHXvZNL/kanban-template



No one can work a second consecutive week of overtime. Even isolated overtime used too frequently is a sign of deeper problems that must be addressed.

Collective Ownership

Every programmer improves any code anywhere in the system at any time if they see the opportunity.

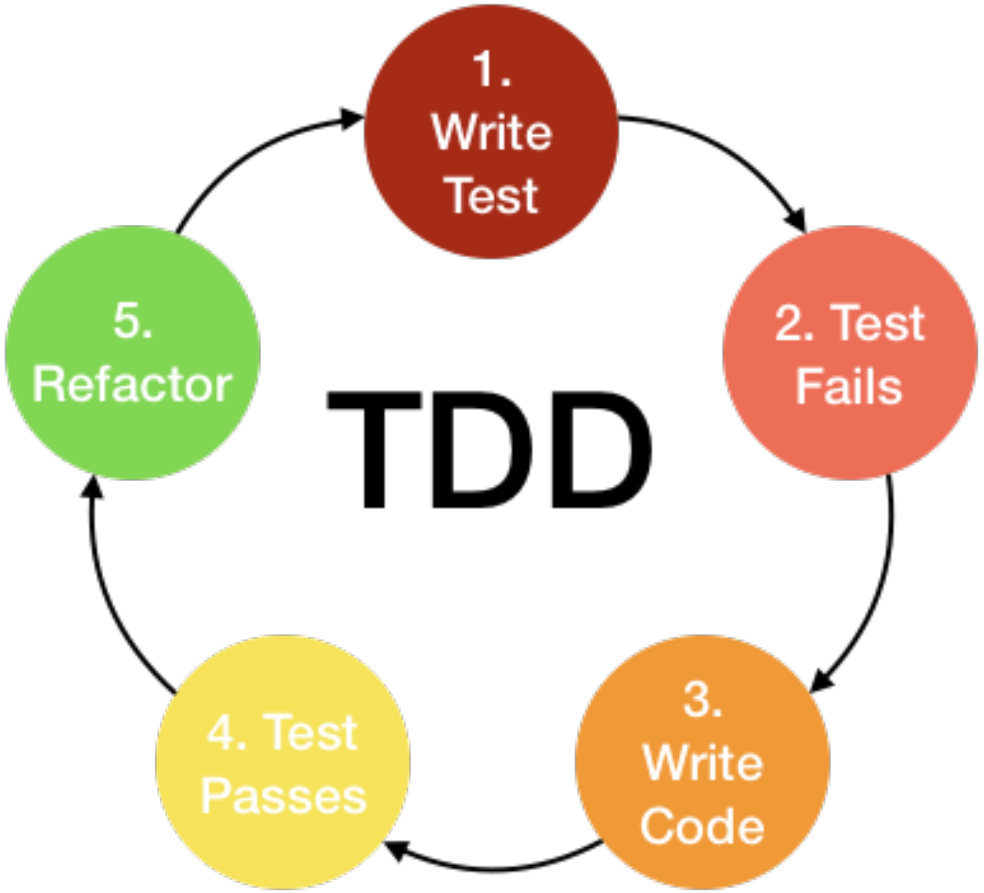


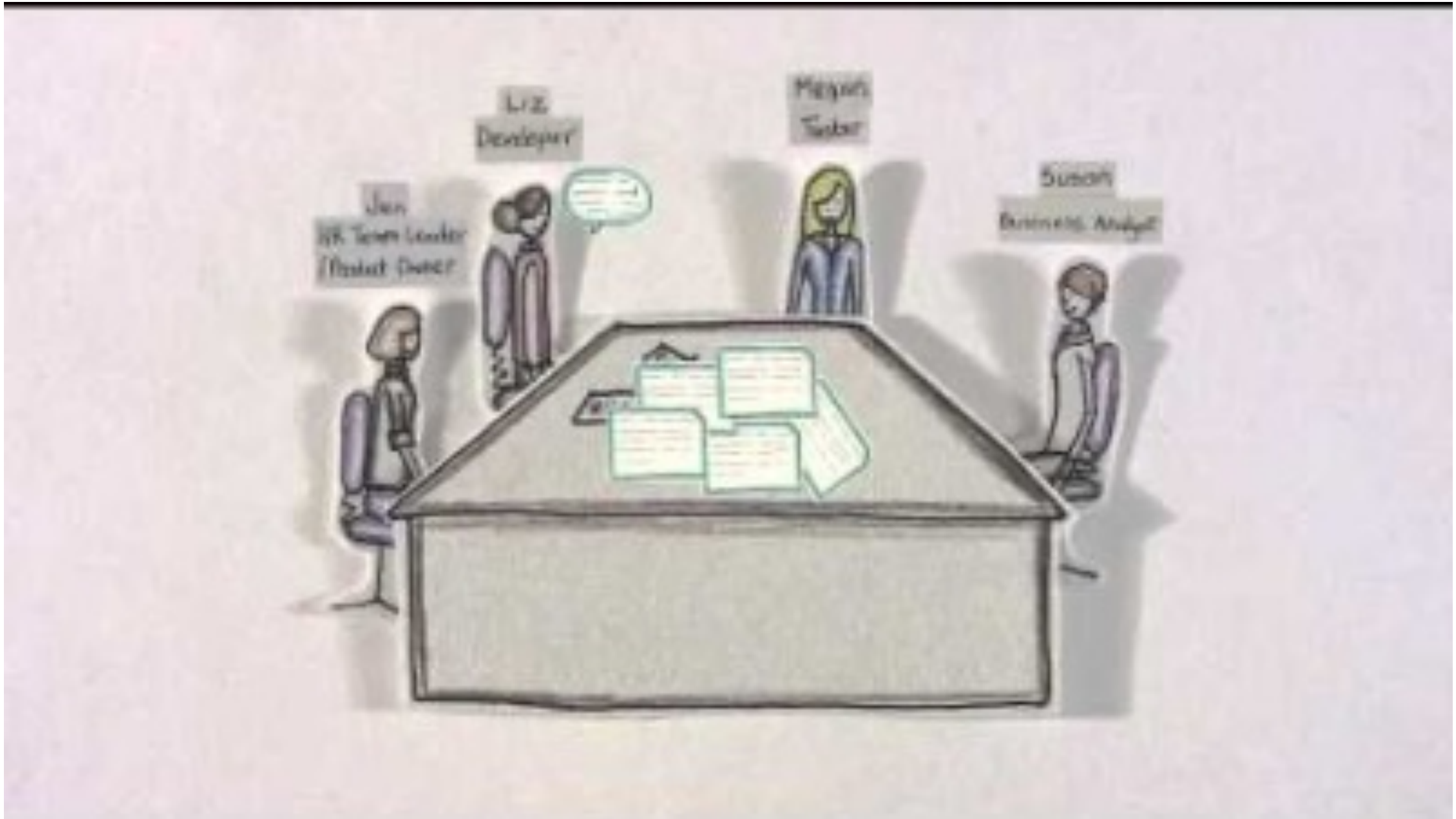
Pair Programming

https://www.youtube.com/watch?v=ET3Q6zNK3Io

Test-driven development

Programmers write unit tests minute by minute. These tests are collected and they must all run correctly. Customers write functional tests for the stories in an iteration.

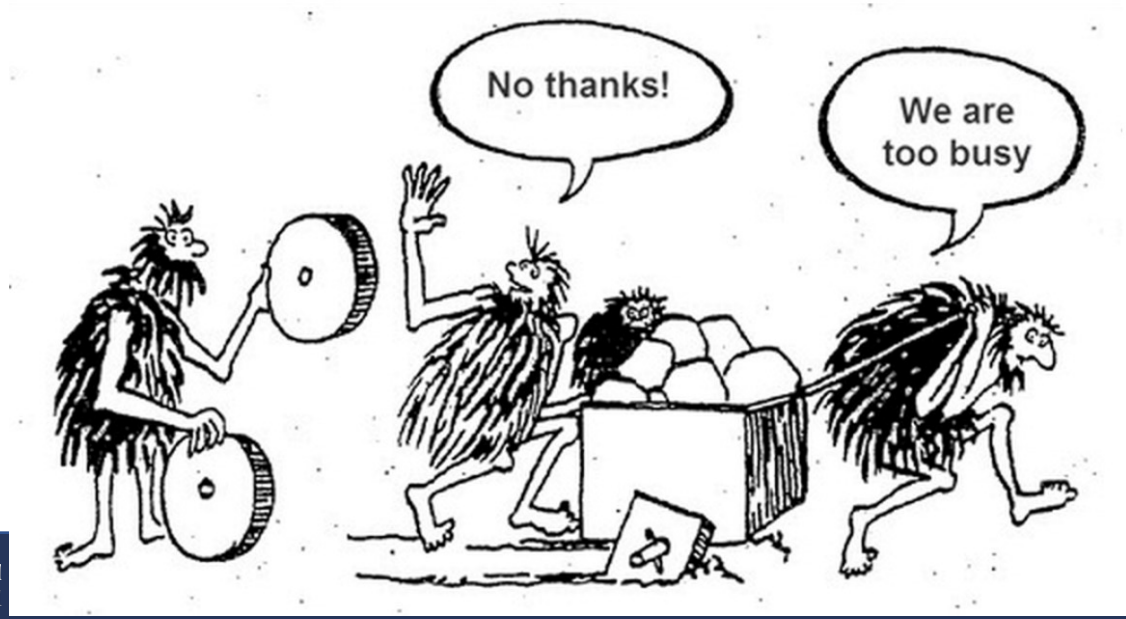


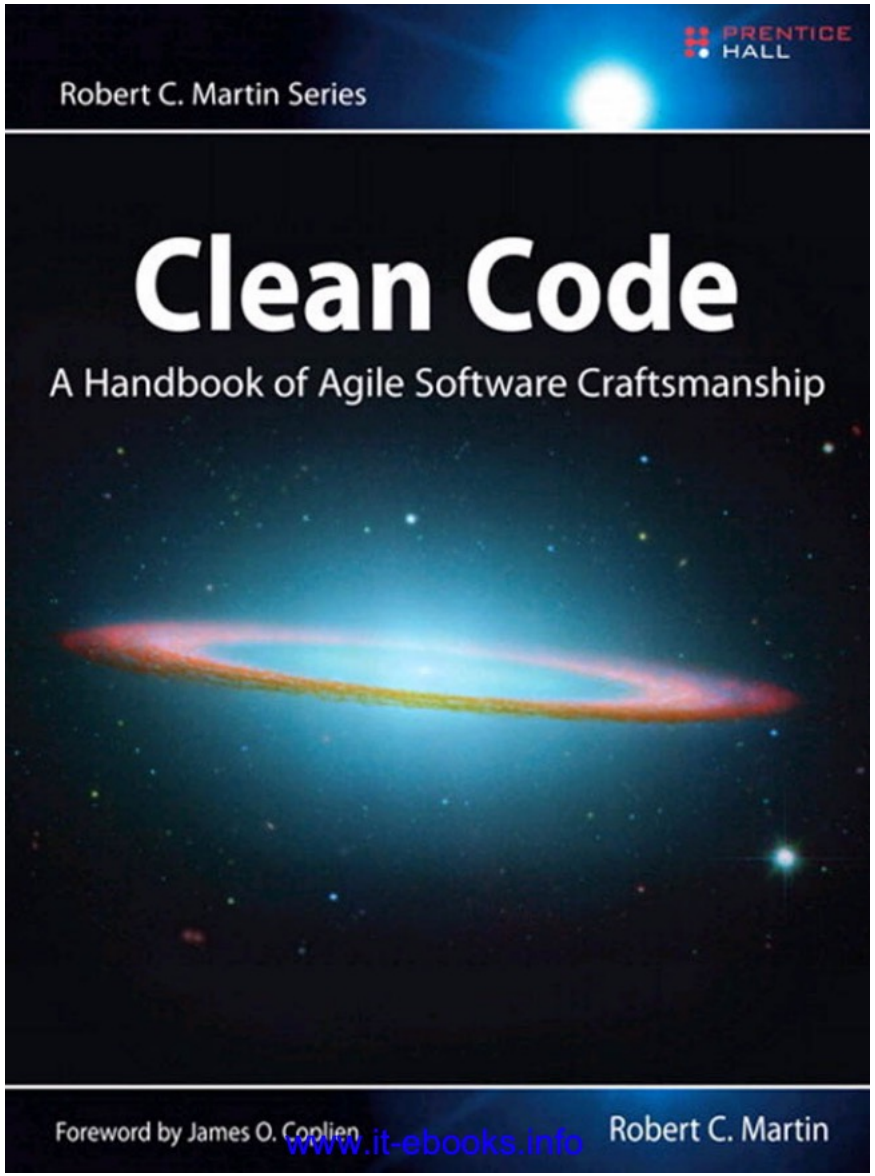


https://www.youtube.com/watch?v=uGaNkTahrIw

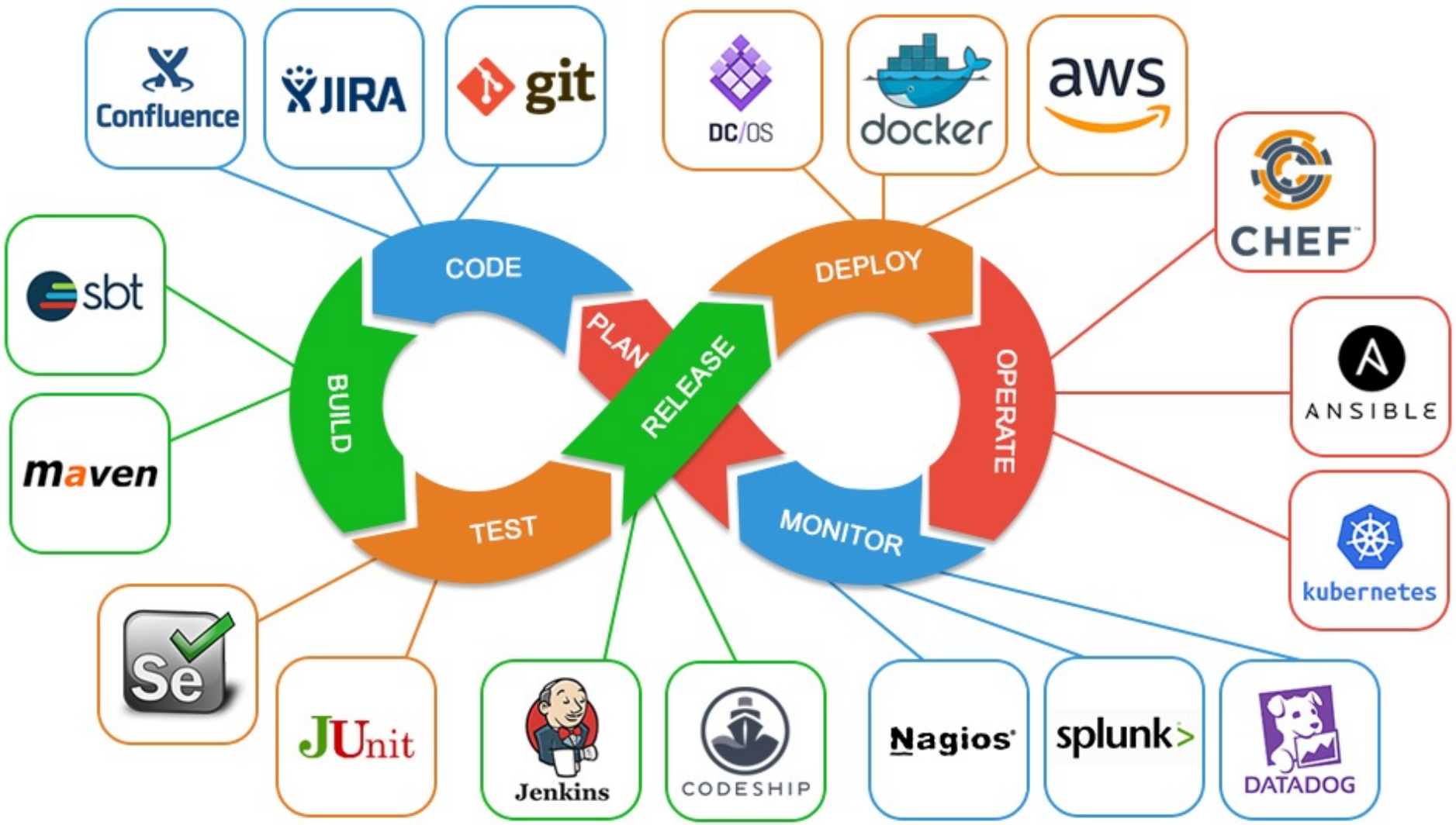
Refactoring vs. Design

The design of the system is evolved through transformations of the existing design that keep all the tests running.





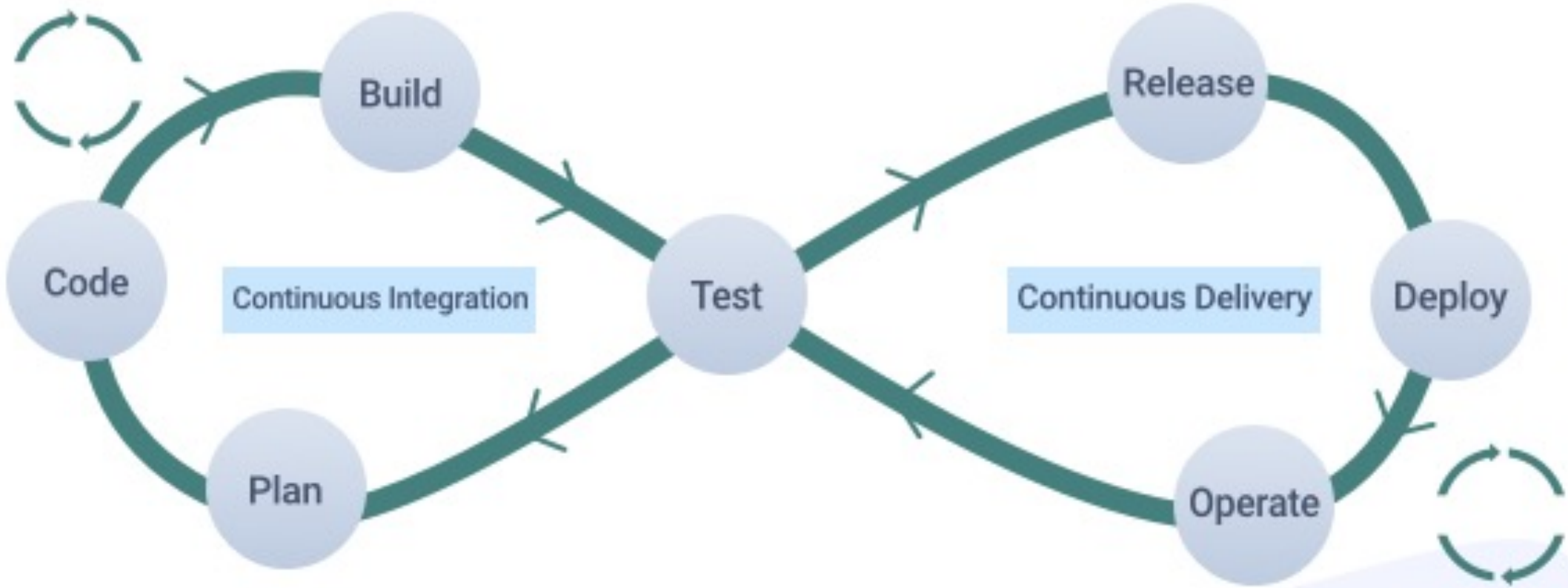
DevOps


Confluence
JIRA
git
DC/OS
docker
aws
CHEF
sbt
maven
ANSIBLE
kubernetes
Se
JUnit
Jenkins
CODESHIP
Nagios
splunk>
DATADOG
CODE
PLAN
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MONITOR

CI/CD

(Continuous Integration/Continuous Delivery)

XENONSTACK



Continuous Integration (CI)

New code is integrated with the current system after no more than a few hours. When integrating, the system is built from scratch and all tests must pass or the changes are discarded.

Continuous Deployment (CD)

is closely related to Continuous Integration and refers to keeping your application deployable at any point or even automatically releasing to a test or production environment if the latest version passes all automated tests.

Apps

Actions

Categories

API management

Chat

Code quality

Code review

Continuous integration ×

Mobile CI

Container CI

Dependency management

Deployment

IDEs

Learning

Localization

Mobile

Monitoring

Project management

Publishing

Recently added

Security

Continuous integration

Automatically build and test your code as you push it to GitHub, preventing bugs from being deployed to production.

1927 results filtered by Continuous integration ×

Sentry Release

By getsentry

GitHub Action for creating a release on Sentry

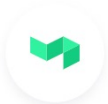
101 stars

Flaptastic

Manage flaky unit tests. Click a checkbox to instantly disable any test on all branches. Works with your current test suite

Buddy

One-click delivery automation for Web Developers



Trigger Buildkite Pipeline

By buildkite

A GitHub Action for triggering a build on a Buildkite pipeline

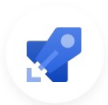
25 stars

Vault Secrets

By hashicorp

A Github Action that allows you to consume HashiCorp Vault™ secrets as secure environment variables

67 stars



Azure Pipelines

Continuously build, test, and deploy to any platform and cloud

Codefresh

A modern container-based CI/CD platform, easily assemble and run pipelines with high performance

CircleCI

Automatically build, test, and deploy your project in minutes

Travis CI

Test and deploy with confidence

GitHub Action for Cloudflare Workers

By cloudflare

Deploy a Cloudflare Worker with the Serverless Framework

89 stars

Scrum Master


scrum police
hero
scribe
secretary
SCRUM MASTER ?
coffee clerk
chairman
JIRA
admin
team boss


impediment remover
facilitator
COACH
coach
teacher
SCRUMMASTER!
servant leader
manager
change agent
mentor

Scrum Master serves the Scrum Team

- Coaching the team members in self-management and cross-functionality;
- Helping the Scrum Team focus on creating high-value Increments that meet the Definition of Done;
- Causing the removal of impediments to the Scrum Team's progress; and,
- Ensuring that all Scrum events take place and are positive, productive, and kept within the timebox.

Scrum Master serves the Product Owner

- Helping find techniques for effective Product Goal definition and Product Backlog management;
- Helping the Scrum Team understand the need for clear and concise Product Backlog items;
- Helping establish empirical product planning for a complex environment; and,
- Facilitating stakeholder collaboration as requested or needed.

Scrum Master serves the organization

- Leading, training, and coaching the organization in its Scrum adoption;
- Planning and advising Scrum implementations within the organization;
- Helping employees and stakeholders understand and enact an empirical approach for complex work; and,
- Removing barriers between stakeholders and Scrum Teams.