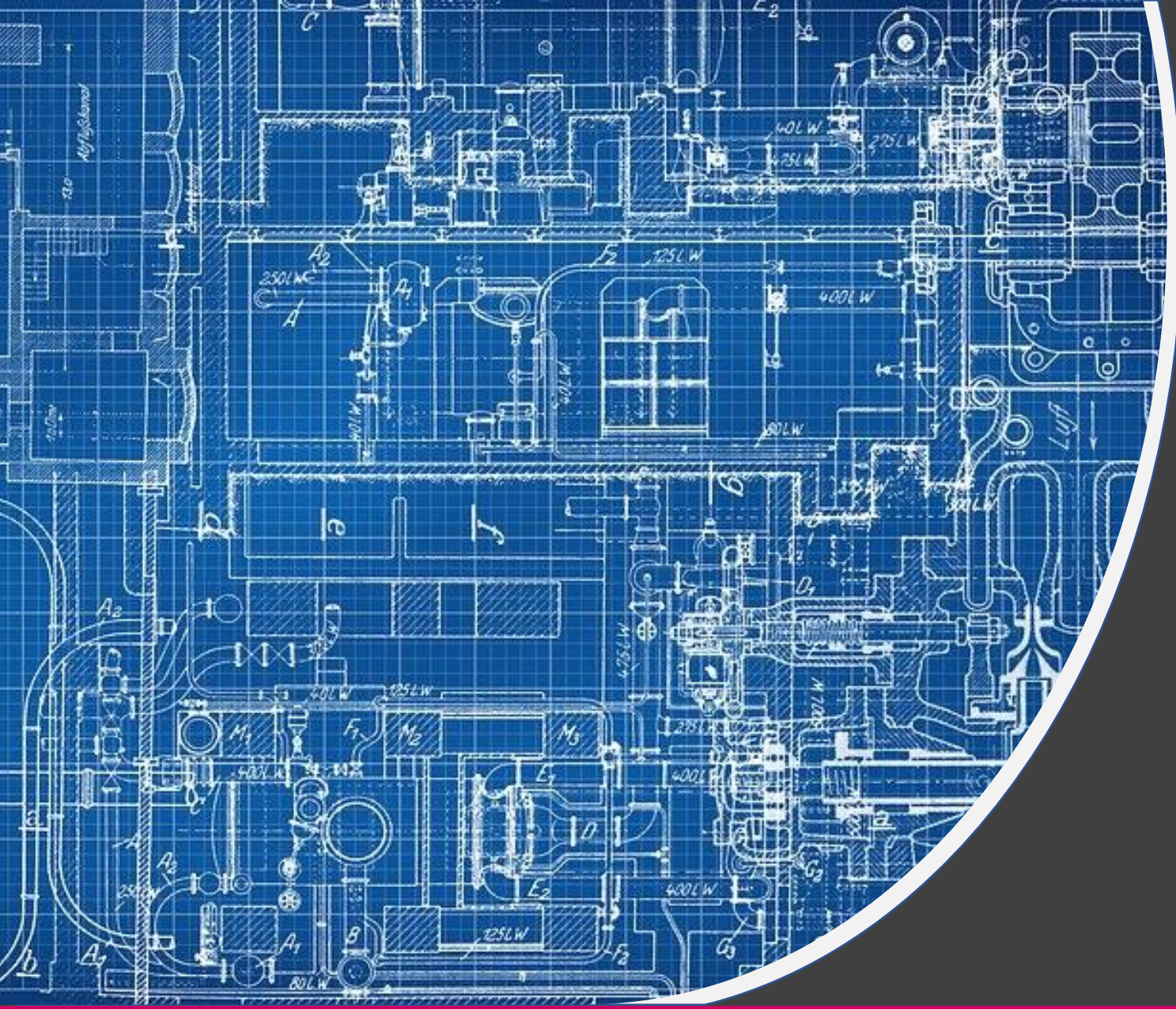


Software Engineering

Objective & expected outcomes

Objective: learn software development techniques to deliver high quality both from a functional and non-function point of view.

By the end of the course, you will be able to:

- Design software that is in line with expected functional and non-functional requirements
- Understand the relationship between architecture, design, implementation and user expectations.
- Implement sophisticated designs and algorithms
- Specify requirements for software systems
- Create maintainable designs and architecture
- Organize a team to execute a medium-sized software project
- Assess / Evaluate design and implementation options
- Choose alternatives to optimize for a given objective (e.g., performance vs maintainability)

Software Engineering?

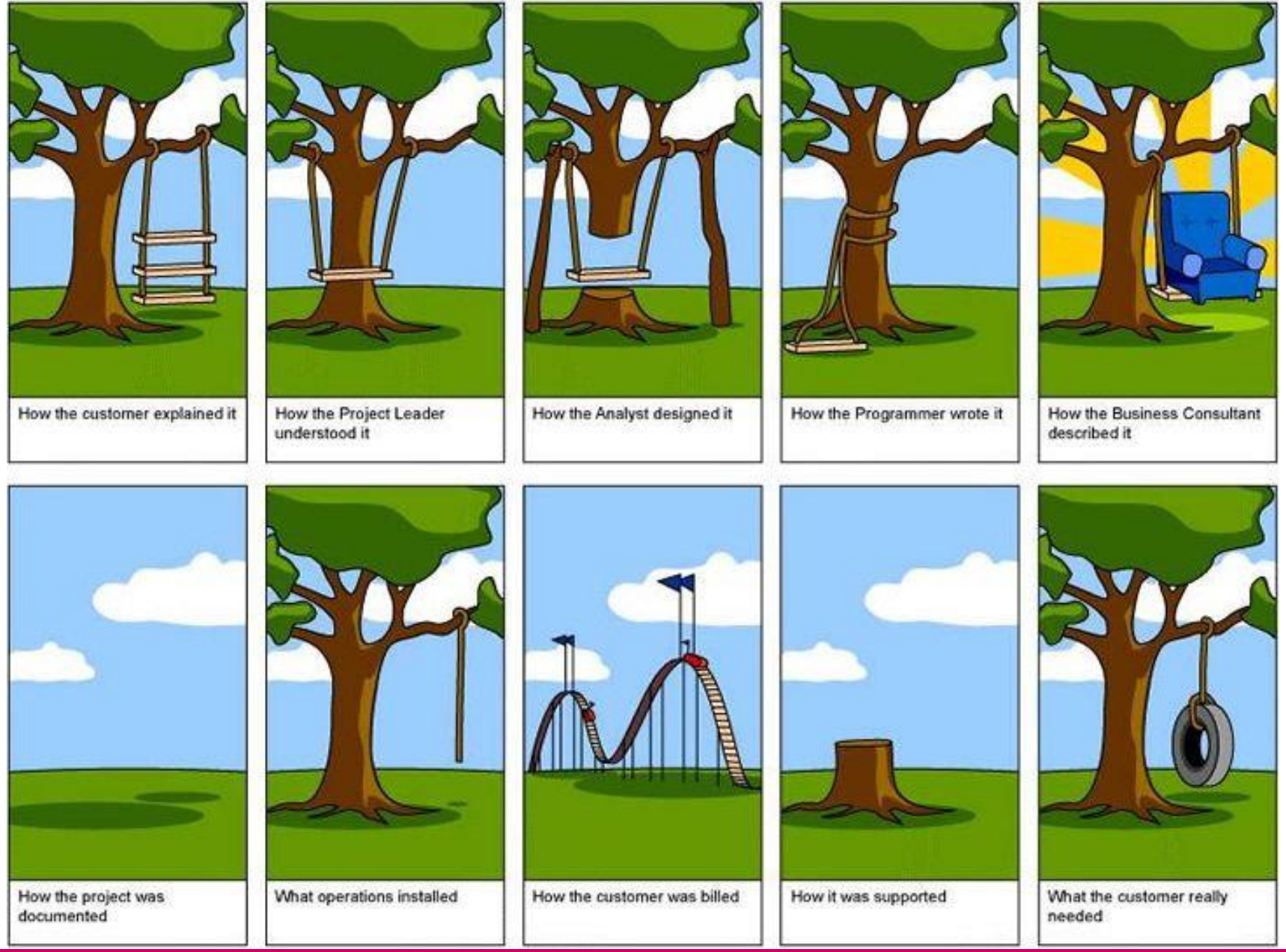
“ **Software engineering** is the systematic application of engineering approaches to the development of software. ”

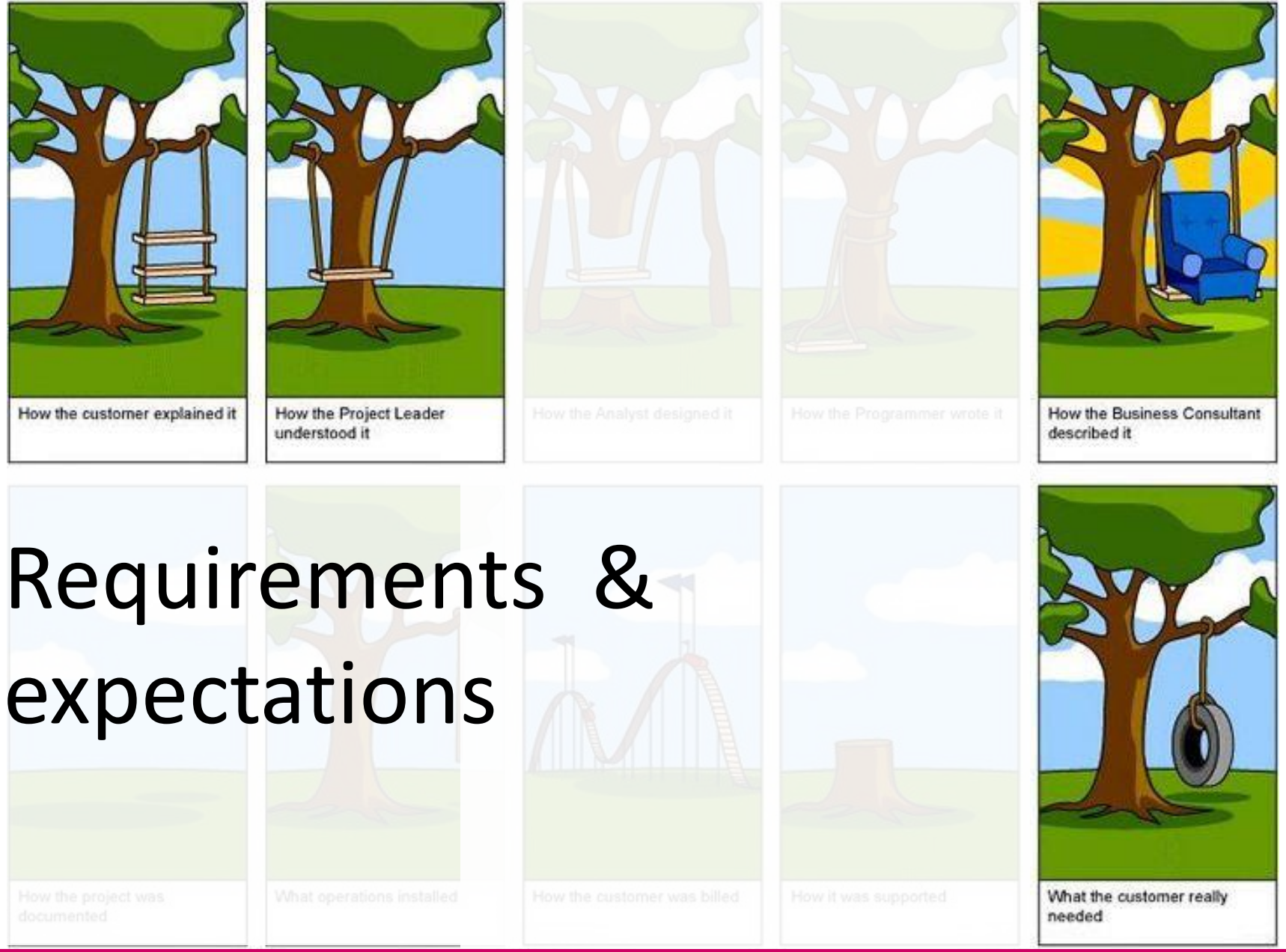
- **Engineering** is the use of scientific principles to design and build machines, structures, and other items, including bridges, tunnels, roads, vehicles, and buildings.
- **Software development** is the process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications, frameworks, or other software components.
- **Computer software**, or simply **software**, is a collection of data or computer instructions that tell the computer how to work.

Software Development Life Cycle

Software development activities and how to organize them

- Requirements: determine the need of the customer
- Design: define the specifications and structure of the software. This step is sometimes split in high-level design (architecture) and detailed design.
- Implementation: actual creation of the software (i.e., coding, configuring)
- Testing: assessing whether the software meets the specifications and the customer expectations
- Maintenance: once in production, improve the software and make sure it remains operational

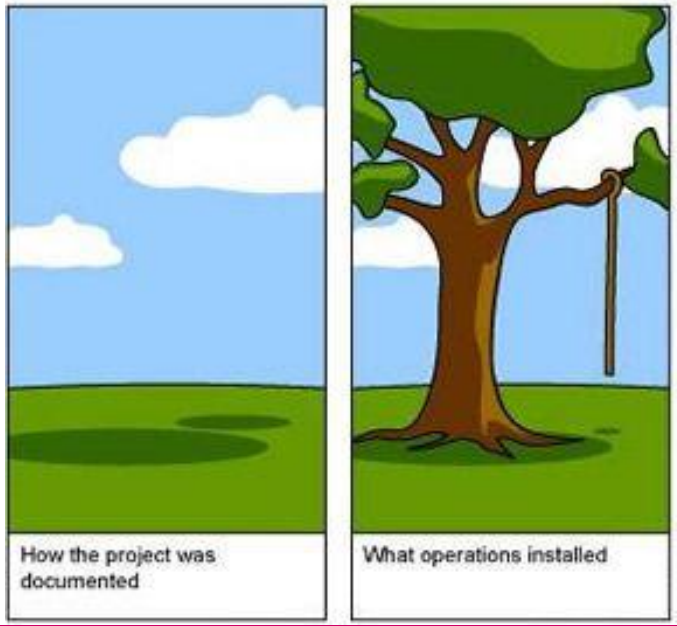

How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed



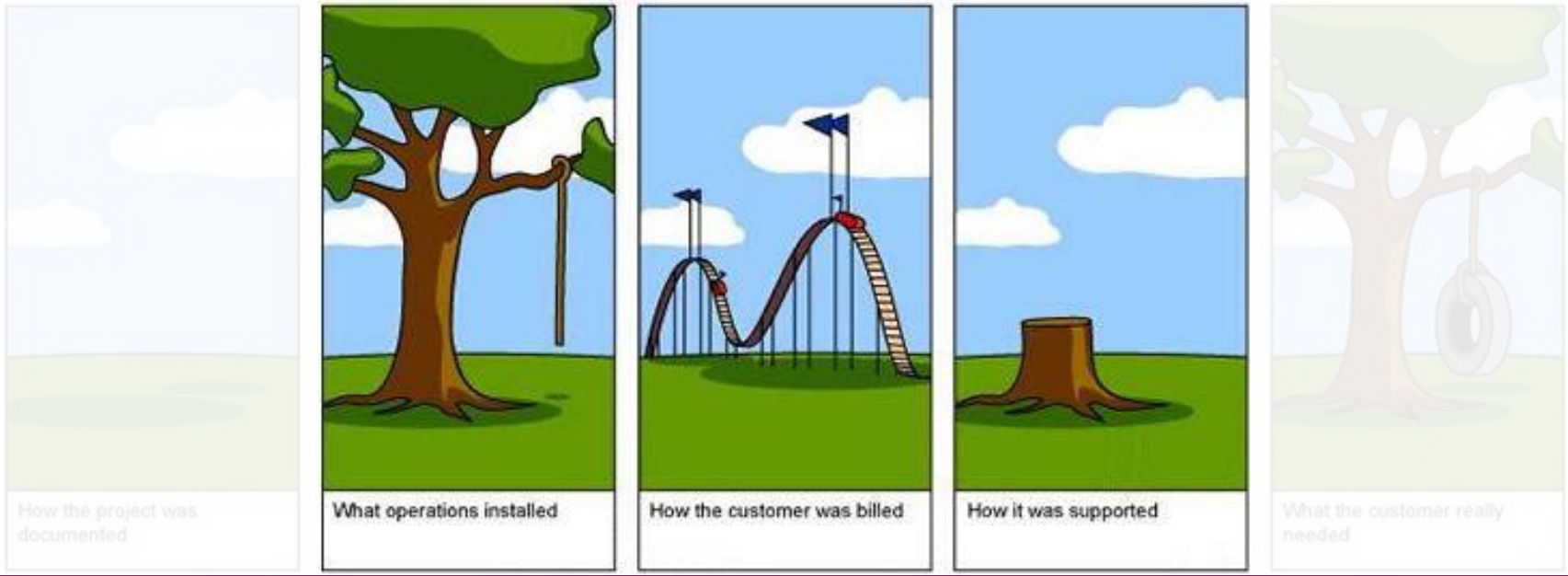
Requirements & expectations

Design and implementation

Testing, Integration, and deployment



Maintenance and operations



Capturing requirements

Listen to requirements, create a high-level design and compare the results between the groups. What do you remark?

Requirements

The need that a stakeholder aims to satisfy:

I want to maximize my chances of winning a legal case

Warning : VERY OFTEN WE MIX THE NEED AND THE SOLUTION TO SATISFY IT

For instance, a search engine that helps to find jurisprudence is the solution to above need. Building a search engine is not the need.

Requirements

What the stakeholders outside of the development team perceive

- Functional (hard to specify and to check)
  - Validity w.r.t. the user requirements
- Non-functional (usually easier to check)
  - Performance: scalability, Throughput
  - Usability: easy to understand, fast to use
  - Re-utilisability: can parts be reused for other software
  - Portability: is it versatile from a platform perspective
  - Interoperability: does it integrate easily with 3rd parties
  - Robustness: Is it resilent when encountering unexpected behaviors

How to evaluate the quality of a requirements?

1. Unitary: the requirements addresses 1 and only 1 need
2. Completeness: the requirements is holistic
3. Consistency: the requirements does not violate other requirements
4. Traceable: the requirement is properly documented and audited
5. Current: the requirements are up to date. Otherwise it leads to scope creep.
6. Unambiguous: requirements is free of technical jargon and is sufficiently documented to not require additional knowledge to be interpreted
7. Priority: requirements is properly prioritized to help optimize the resources
8. Verifiable: the requirement's implementation can be verified and there is clear way of validating whether it has been properly implemented

How to capture requirements?

High-level / Pitching

DOCUMENT VISION

LEAN CANVAS

Detailed

USER STORIES / USE CASES

STORYBOARDS

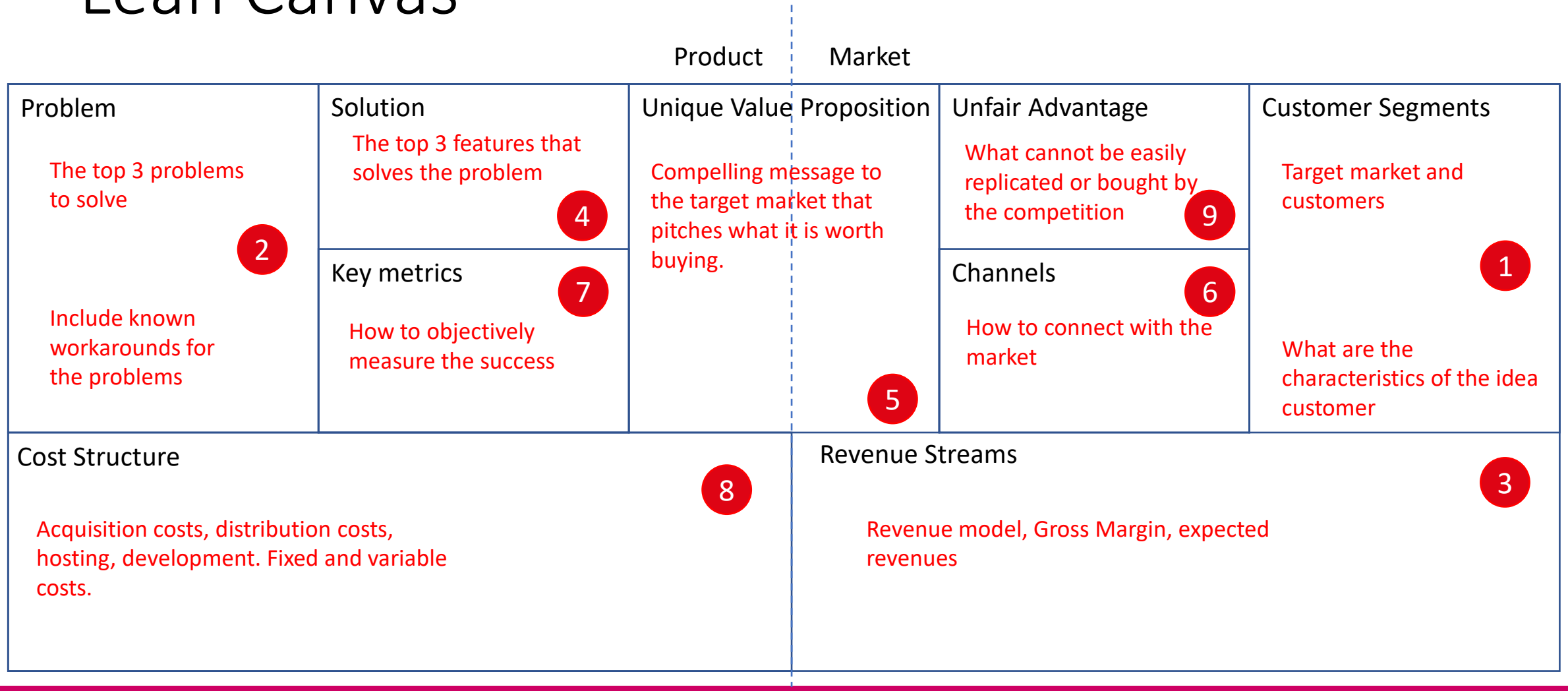
NON-FUNCTIONAL

Vision Document

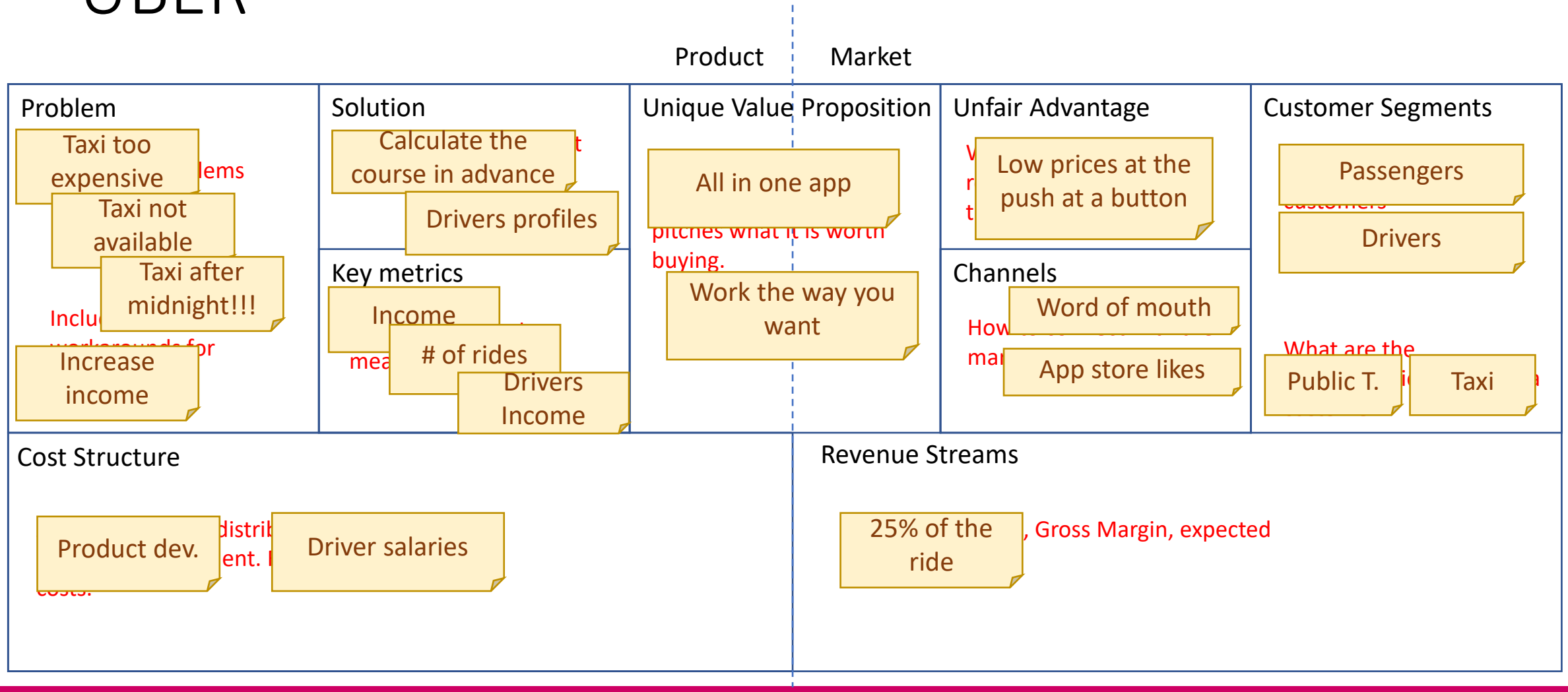
- To establish a common vision of the project amongst the stakeholders
- Define scope, problem statement, major features, competition, marketing…

Template : https://docs.google.com/document/d/1tj51zARj37kZ8gC70EiJUAyW-mPGL2qVTfEPeKbtSgk/edit

Lean Canvas


Product
Market
Problem
The top 3 problems to solve
2
Include known workarounds for the problems
Solution
The top 3 features that solves the problem
4
Key metrics
7
How to objectively measure the success
Unique Value Proposition
Compelling message to the target market that pitches what it is worth buying.
5
Unfair Advantage
What cannot be easily replicated or bought by the competition
9
Channels
6
How to connect with the market
Customer Segments
Target market and customers
1
What are the characteristics of the idea customer
Cost Structure
8
Acquisition costs, distribution costs, hosting, development. Fixed and variable costs.
Revenue Streams
3
Revenue model, Gross Margin, expected revenues

UBER


Product
Market
Problem
Taxi too expensive
Taxi not available
Taxi after midnight!!!
Increase income
Solution
Calculate the course in advance
Drivers profiles
Key metrics
Income
# of rides
Drivers Income
Unique Value Proposition
All in one app
Work the way you want
Unfair Advantage
Low prices at the push at a button
Channels
Word of mouth
App store likes
Customer Segments
Passengers
Drivers
Public T.
Taxi
Cost Structure
Product dev.
Driver salaries
Revenue Streams
25% of the ride

Lean Canvas

Establish the lean canvas for your great idea. Get ready to present it next week.

User stories

As a <type of user>, I want to <some goal> so that <reason to achieve the goal>

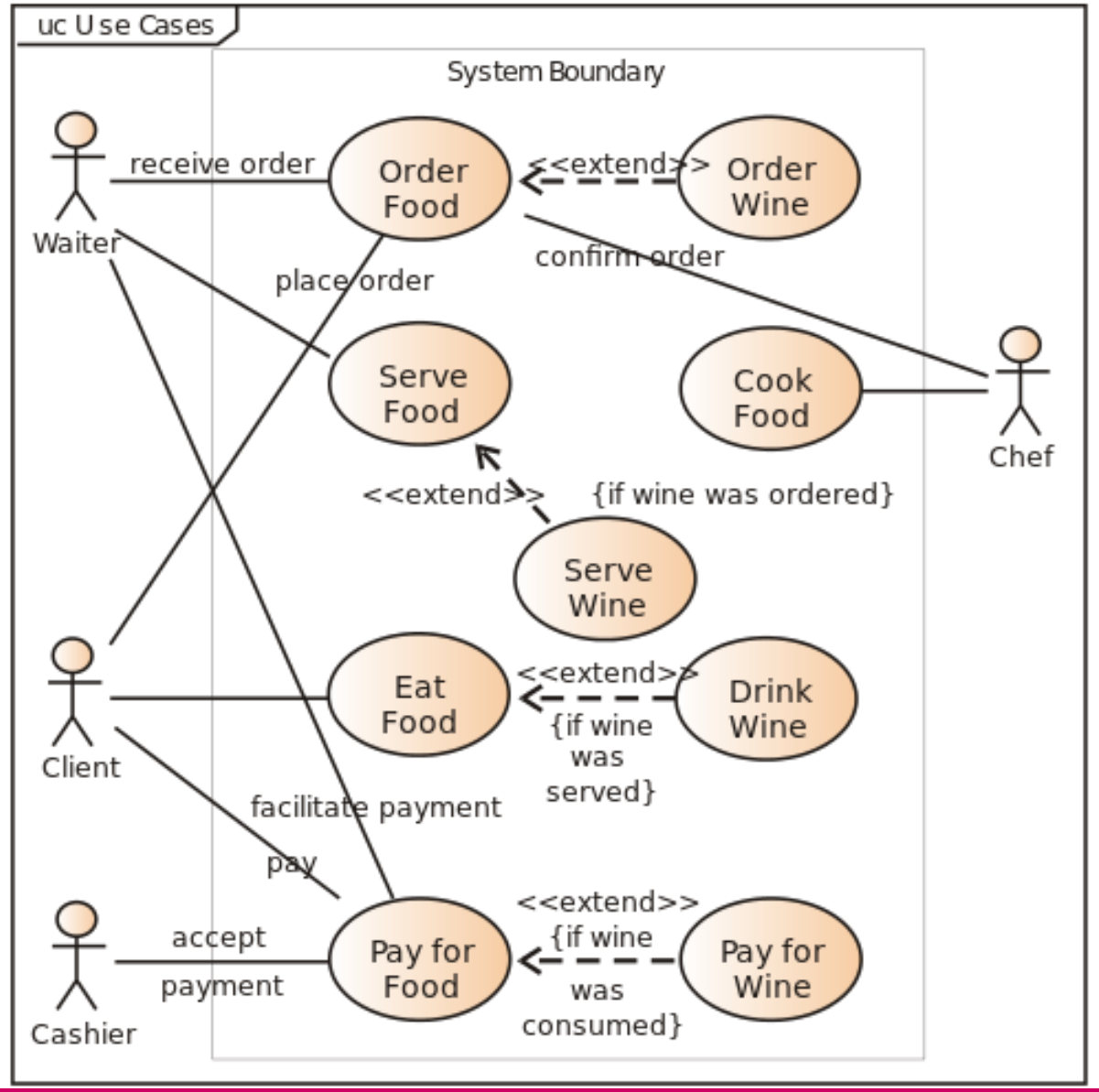
As a visitor in a foreign country, I want to be able to book a cab in one click knowing the cost in advance so that I don't get screwed because I do not know the local customs.

User Stories

Create stories for the feature for services like last pass, onepass

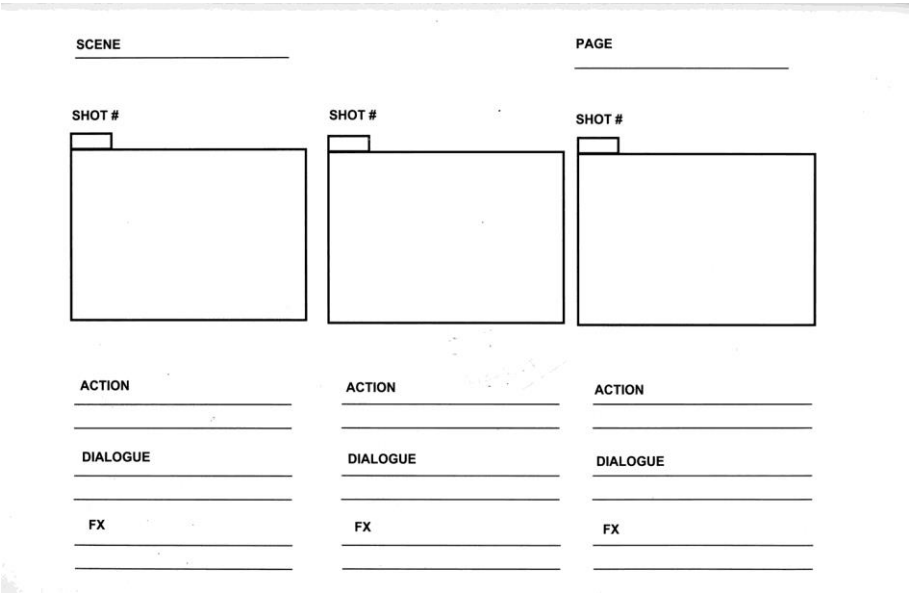
Use Cases

- Use case is a subset of the Unified Meta Language
- Describes the interactions between a role a.k.a. an actor and the system to achieve a particular goal.



Storyboard

Tell a story with a sequence of drawings/sketches/images



Storyboards

Create the storyboards for your project

Non-Functional requirement template

Non-Functional Requirements (NFR) defines how the system operates NOT what it does. Also called quality attributes:

- Execution quality attributes
  - security, usability, performance (throughput, response time), fault tolerance, high availability, auditability….

  Availability

- Evolution quality attributes
  - Maintainability, testability, interoperability, performance (scalability)

NFR template : https://www2a.cdc.gov/cdcup/library/templates/CDC_UP_Non-Functional_Requirements_Definition_Template.doc

Non-Functional Requirements

Establish the non-functional requirements based on the template for your project