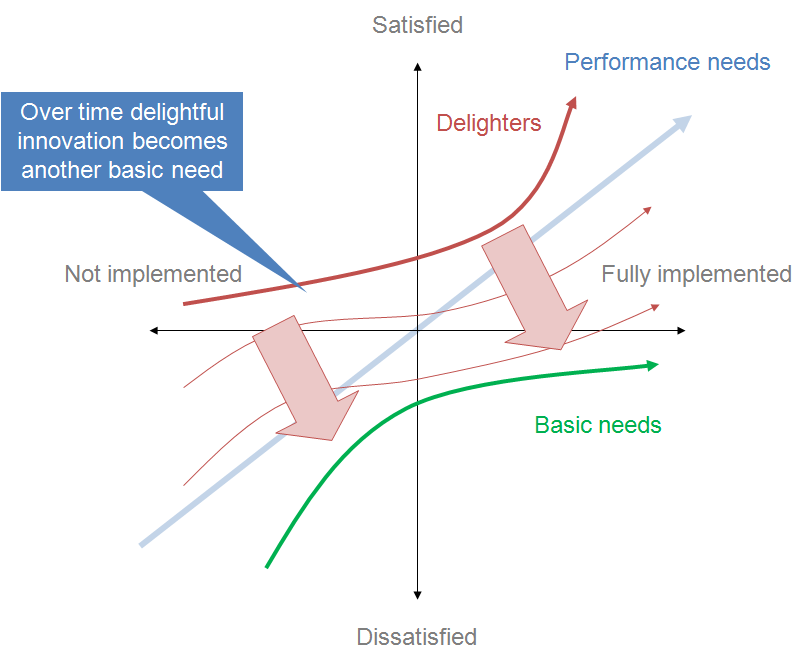
# Product Requirements

## Kano-Model (B)

There are different levels of requirement or qualities:

* **Must-be qualities**: These are qualities the customer expects. If they are not achieved, it leads to unhappiness.
* **One-dimensional qualities**: These attributes result in satisfaction when fulfilled and dissatisfaction when not fulfilled.
* **Attractive Quality**: These attributes provide satisfaction when achieved fully, but do not cause dissatisfaction when not fulfilled.
* **Indifferent Quality**: These attributes refer to aspects that are neither good nor bad, and they do not result in either customer satisfaction or customer dissatisfaction.
* **Reverse Quality**: These attributes refer to a high degree of achievement resulting in dissatisfaction and to the fact that not all customers are alike.

Over time the attractive qualities become basic and are nothing special any more.



### Example

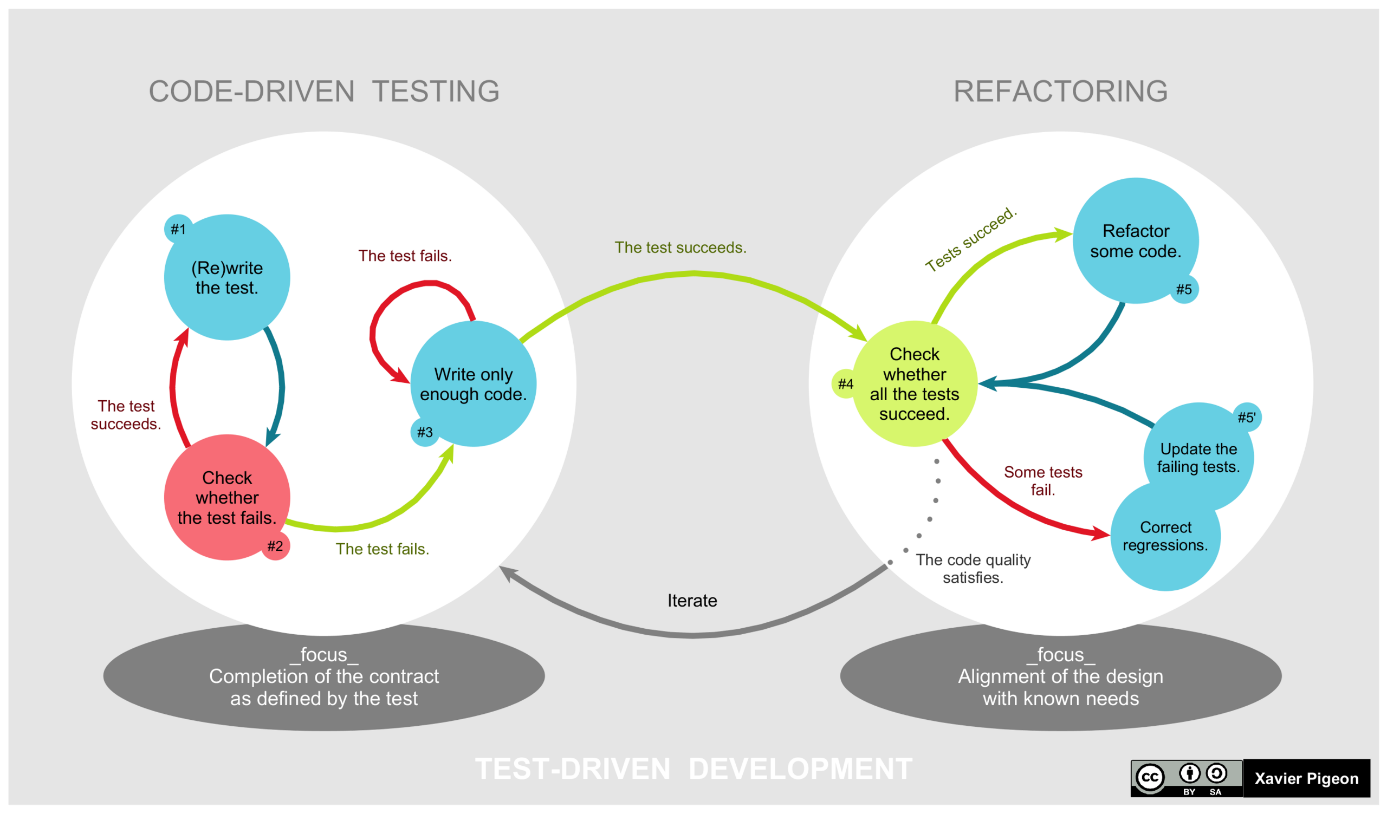
As an example, I chose a digital pen.

|  |  |
| --- | --- |
| Must-be qualities | Write on supported devices |
| One-dimensional qualities | Long battery life – short battery life |
| Attractive qualities | Tilt mechanism for shading and benefit from increased sensitivity and barely noticeable time delay |
| Indifferent qualities | Magnetic fastening on right / left side of laptop |
| Reverse qualities | No further support for future laptops |

## Test-Driven Development (A)

The big difference to normal project management is that test-driven developments has to think about requirements before writing code rather than after. Then run all tests, and the new tests should fail because otherwise no new code would be needed. It rules out the possibility that the new test is flawed and will always pass.

Write enough code until all tests pass. Then refactor the code (only small segments) and run the test to see if functionality isn’t broken.



## Evaluate Product Requirements (C)

* Completeness
* Correctness
* Free of contractions
* Consistency
* Feasibility

### Product Requirements Document / Product Roadmap Document (PRD)

A product requirements document is an official instrument used to convey the features and functionalities of a new product.

Also known as a product roadmap document, PRDs lay out a diverse range of characteristics, requirements, and specifications along with terms and conditions for the design and development stages. Product roadmaps:

* Allow you to get a big picture of a prospective product.
* Estimate the cost of each step and all the important tasks of the development process.
* Give you an indication of how long it takes you to complete key product goals.
* Promote a sense of unity among the team members.
* Mitigate some of the problems that arise throughout development.

A product roadmap does not only highlight the primary objective and unique value proposition of a product but its timeline and future goals as well. This enables the business to craft a product strategy that focuses on delivering customer benefits.

### Components of a PRD

#### Primary Purpose

This describes the high-level goals and objectives of the proposed product to set the strategic direction. It tells all the stakeholders what they need to accomplish at the end of development.

#### Features

This is where product managers highlight all the features in detail with a clear use case, requirements, and context. The more complex your features are, the more you must incorporate bullet points and sub-headings to make them more approachable for all the teams.

#### Basic Design Guidance

The product design team must provide general instructions and guidance on how developers must approach the UI and UX of the product.

#### System and Environment Specifications

To write an effective product roadmap, you must include all the end-user systems that the proposed product must be compatible with. This includes operating systems, and browsers, among others.

#### Estimated Timeline

Without a timeline, product roadmaps remain incomplete. You should have a target release window along with major project milestones and release dependencies.

## FMEA (C)

You have to ensure that the product meets the conditions / requirements in the end. However, the product has to go through lots of different sectors in the company (production, development, design, programming, …).

FMEA (Failure Mode and Effects Analysis) gives qualitative outcomes that you can work with.

### Benefits

* It provides a documented method for selecting a design with a high probability of successful operation and safety.
* A documented uniform method of assessing potential failure mechanisms, failure modes and their impact on system operation, resulting in a list of failure modes ranked according to the seriousness of their system impact and likelihood of occurrence.
* Early identification of single failure points (SFPS) and system interface problems, which may be critical to mission success and/or safety. They also provide a method of verifying that switching between redundant elements is not jeopardized by postulated single failures.
* An effective method for evaluating the effect of proposed changes to the design and/or operational procedures on mission success and safety.
* A basis for in-flight troubleshooting procedures and for locating performance monitoring and fault-detection devices.
* Criteria for early planning of tests.

### How to do?

Give values to each property per item:

#### Probability

How likely will the failure occur?

|  |  |
| --- | --- |
| **Rating** | **Meaning** |
| 1 | Extremely unlikely (virtually impossible or No known occurrences on similar products or processes, with many running hours) |
| 2 | Remote (relatively few failures) |
| 3 | Occasional (occasional failures) |
| 4 | Reasonably possible (repeated failures) |
| 5 | Frequent (failure is almost inevitable) |

#### Severity

How bad effects would this failure have?

|  |  |
| --- | --- |
| **Rating** | **Meaning** |
| 1 | No relevant effect on reliability or safety |
| 2 | Very minor, no damage, no injuries, only results in a maintenance action (only noticed by discriminating customers) |
| 3 | Minor, low damage, light injuries (affects very little of the system, noticed by average customer) |
| 4 | Critical (causes a loss of primary function; loss of all safety margins, 1 failure away from a catastrophe, severe damage, severe injuries, max 1 possible death) |
| 5 | Catastrophic (product becomes inoperative; the failure may result in complete unsafe operation and possible multiple deaths) |

#### Detection

How likely will a maintainer detect this possible failure?

|  |  |
| --- | --- |
| **Rating** | **Meaning** |
| 1 | Certain – fault will be caught on test – |
| 2 | Almost certain |
| 3 | High |
| 4 | Moderate |
| 5 | Low |
| 6 | Fault is undetected by operators or maintainers |

#### Risk level (P×S) and (D)

Risk is the combination of end effect probability and severity where probability and severity includes the effect on non-detectability (dormancy time).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Severity**  **Probability** | **I** | **II** | **III** | **IV** | **V** | **VI** |
| I | Low | Low | Low | Low | Moderate | High |
| II | Low | Low | Low | Moderate | High | Unacceptable |
| III | Low | Low | Moderate | Moderate | High | Unacceptable |
| IV | Low | Moderate | Moderate | High | Unacceptable | Unacceptable |
| V | Moderate | Moderate | High | Unacceptable | Unacceptable | Unacceptable |

After this step FMEA has become a FMACE (failure mode, effects, and criticality analysis).

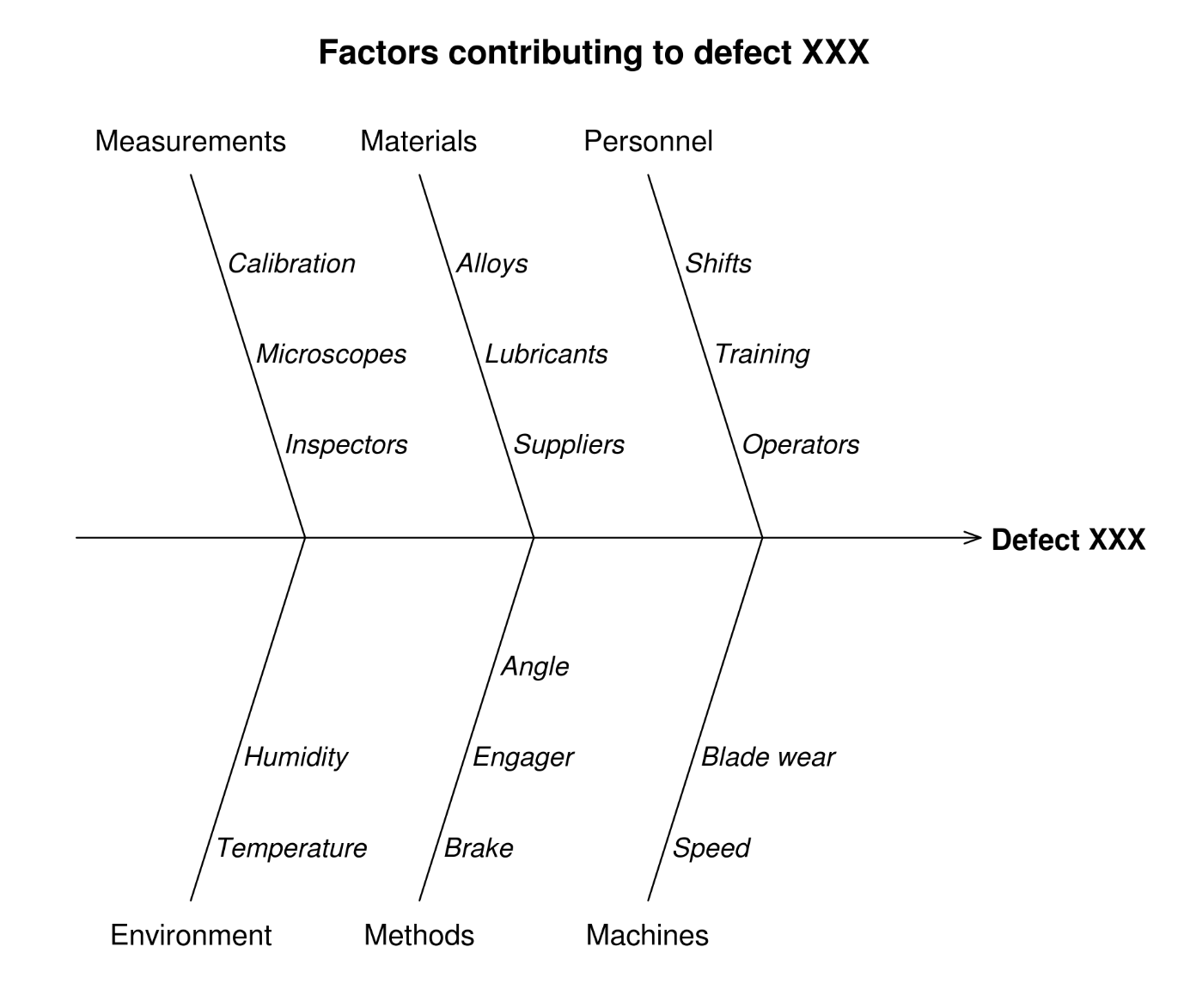
There are even more factors which should be given values. In the end you have worksheet which gives you a perfect overall and also details picture of **potential failures of components, assemblies, and subsystems and their effects**.

### Advantages

* Catalyst for teamwork and idea exchange between functions
* Collect information to reduce future failures, capture engineering knowledge
* Early identification and elimination of potential failure modes
* Emphasize problem prevention
* Fulfill legal requirements (product liability)
* Improve company image and competitiveness
* Improve production yield
* Improve the quality, reliability, and safety of a product/process
* Increase user satisfaction
* Maximize profit
* Minimize late changes and associated cost
* Reduce impact on company profit margin
* Reduce system development time and cost
* Reduce the possibility of same kind of failure in future
* Reduce the potential for warranty concerns

## Ishikawa-Diagram / Fishbone-Diagram (D)

This diagram shows you the potential causes of a specific event. Each cause or reason for imperfection is a source of variation. Causes are usually grouped into major categories to identify and classify these sources of variation. The purpose of this diagram is to find and, as a result, prevent potential root causes. Therefore, you can control the quality of products.



### Root causes

The main goal is to find the root causes of any problem of quality for example in manufacturing or production. Because if you can eliminate the root cause, the problems are less likely to or will never occur again.

In most scenarios these groups / categories / big fish bones are grouped by the following categories: However the fishbone categories can be unique for specific use cases.

### The 5 Ms for manufacturing

* Manpower / mind power (physical or knowledge work, includes: kaizens, suggestions)
* Machine (equipment, technology)
* Material (includes raw material, consumables, and information)
* Method (process)
* Measurement / medium (inspection, environment)

Or 8 Ms:

* Mission / mother nature (purpose, environment)
* Management / money power (leadership)
* Maintenance

### The 8 Ps for product marketing

* Product (or service)
* Price
* Place
* Promotion
* People (personnel)
* Process
* Physical evidence (proof)
* Performance

### The 4 or 5 Ss for service industries

* Surroundings
* Suppliers
* Systems
* Skill
* Often an important 5th S is added – Safety

TODO: Find out about how to evaluate product requirements!