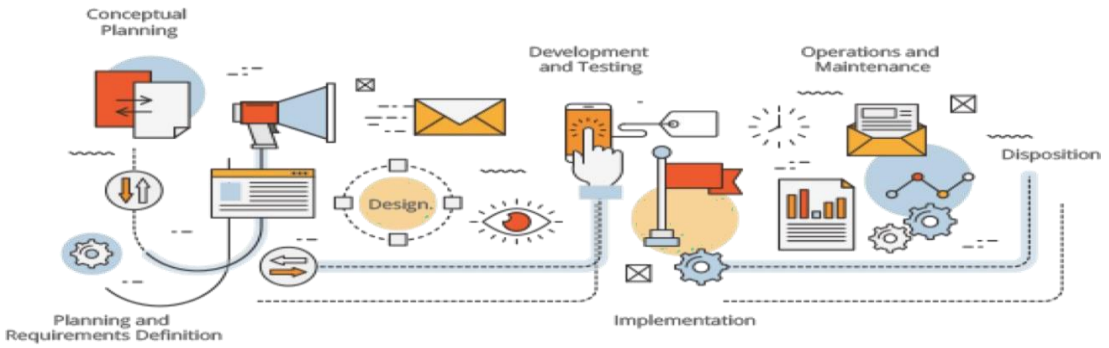
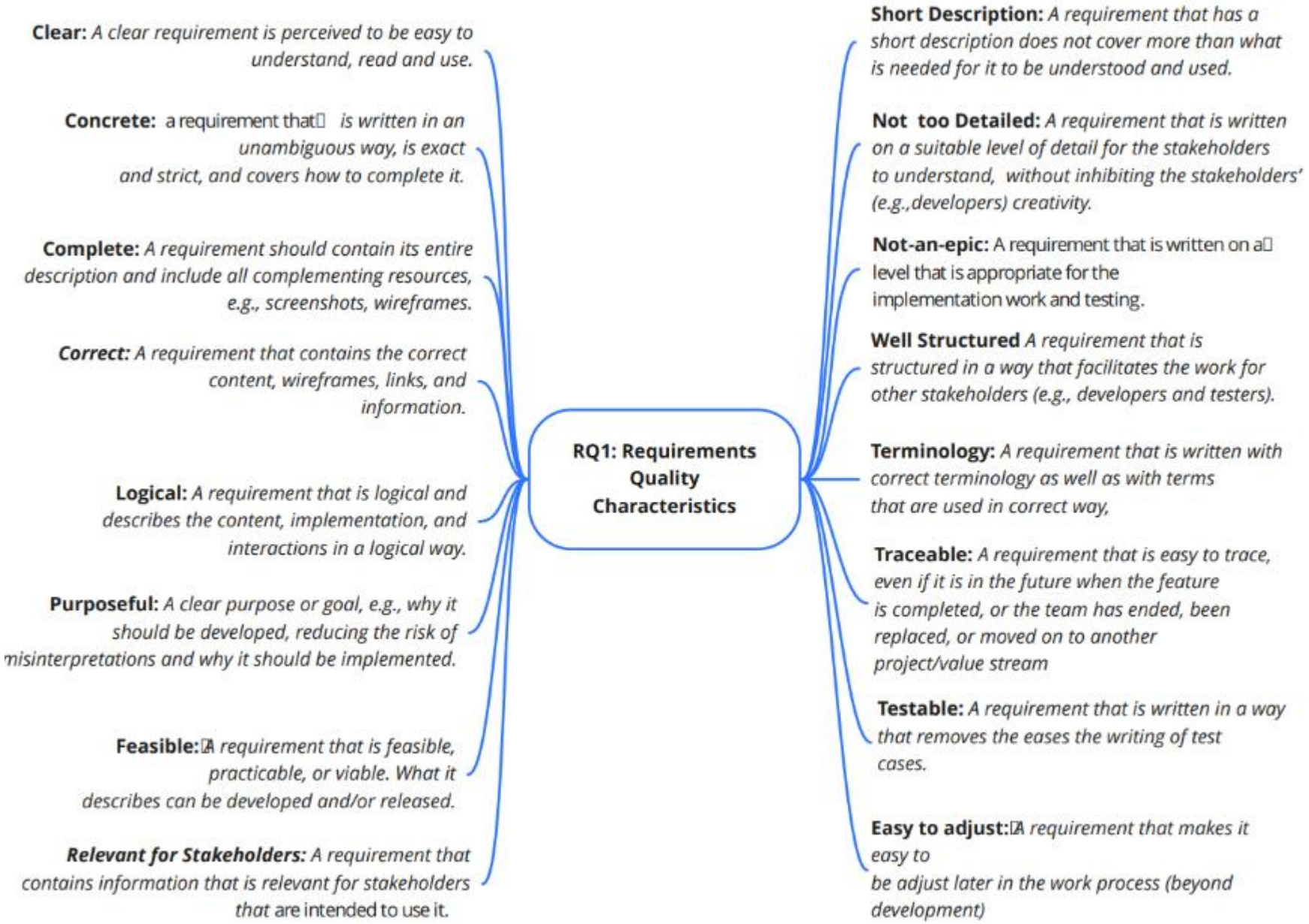
Software Engineering

Requirement Engineering



Attributes of a good requirement



The system shall never fail.

Reason: Impracticality, Lack of Measurability, Ambiguity and Unrealistic Expectations

"The system shall maintain a 99.99% uptime during standard operating hours, prioritizing graceful degradation over complete outages. Critical functionalities must sustain a minimum 99.9% uptime.

"The software should prioritize both security and flexibility."

Reason: This requirement presents a potential **conflict** between security and flexibility priorities. Security measures (e.g., strict access controls, encryption) may sometimes limit flexibility (e.g., restricting user actions), while increased flexibility (e.g., customizable user permissions) may introduce security vulnerabilities. Without clarification or prioritization of which aspect is more important in different scenarios, developers may face difficulties in designing and implementing the system.

The system shall work with any browser.

Reason: The requirement for universal browser compatibility poses significant challenges. lacks specific criteria or metrics to define what it means for the system to "work" with a browser. Without clear benchmarks. t's unclear whether the requirement refers to all existing browsers. Huge scope results in extending the deadline.

"The system shall be compatible with major modern browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge, ensuring consistent functionality and user experience across platforms."

"The system should be efficient in managing library operations.“

Reason: This requirement lacks measurability as "efficient" is a subjective term. Without specific metrics or criteria for measuring efficiency (e.g., processing time for book checkouts, turnaround time for book returns, queue management for book reservations), it's challenging for developers to objectively assess whether the system meets this requirement.

The system should improve administrative processes.

Issues: This requirement is **overly broad and lacks specificity**. It does not specify which administrative processes need improvement or how improvement will be measured. Without clear objectives and metrics, it's challenging for developers to understand and implement this requirement effectively.

"The system should provide real-time updates."

Improved Requirement:
"The system shall deliver real-time updates to flight crews, ground staff, and passengers regarding flight status, gate changes, and schedule adjustments. It shall support multiple communication channels, including mobile applications, SMS, email, and public address systems, to ensure timely dissemination of information."

The system must have good usability.

The term "good usability" is subjective and can be interpreted differently by different stakeholders. lacks clarity regarding specific usability criteria or characteristics that define "good usability."

“The system shall be designed to prioritize user experience by incorporating intuitive navigation, adherence to accessibility standards, and responsive design elements.”

The user should not have to wait a long time for the transaction.	Not measurable. Better would be: The software shall complete the transaction within one second.
The software shall initialize all variables before use.	Not externally verifiable. This is a coding style, which may be part of the SOPs, but it is not a software requirement of the system.
The software shall monitor the ambient temperature.	Not externally verifiable. There is no reason for the software to monitor something but not do something with the data, so put what gets done (display, errors…).

- Requirements must do **ONE THING.**
  - Bad:
    - The system shall accept credit cards and accept pay pal
  - Good:
    - The system shall accept credit cards
    - The system shall accept pay pal.

- • Requirements must be **testable**. Use precise language.
  - Bad:
    - The system shall work with any browser
  - Good:
    - The system shall work with Firefox
    - The system shall work with IE
  - Bad:
    - The system shall respond quickly to user clicks
  - Good:
    - The system shall respond within 10ms to any user click

“A user is to be given access to the system instantaneously after submitting a request to sign on.”

- A poor requirement such as this is very common. The words “instant”, “instantaneously”, “immediate” and “immediately” should always be avoided.

Ambiguous Terms	Ways to Improve Them
at least, at a minimum, not more than, not to exceed	Specify the minimum and maximum acceptable values.
between X and Y	Define whether the end points are included in the range.
efficient	Define how efficiently the system uses resources, how quickly it performs specific operations, or how quickly users can perform certain tasks.
fast, quick, rapid	Specify the minimum acceptable time in which the system performs some action.
flexible, versatile	Describe the ways in which the system must be able to adapt to changing conditions, platforms, or needs.
i.e., e.g.	People often confuse i.e. (meaning "that is") and e.g. (meaning "for example"). Avoid Latin abbreviations.
improved, better, faster, superior	Quantify how much better or faster constitutes adequate improvement in a specific functional area.
including, including but not limited to, and so on, etc., such as, for instance	List all possible values or functions or point to the location of the full list so all readers know what the whole set of items contains.
in most cases, generally, usually, almost always	Clarify when the stated conditions do not apply and what happens then.
maximize, minimize, optimize	State the maximum and minimum acceptable values of some parameter.
normally, ideally	Identify abnormal or non-ideal conditions and describe how the system should behave then.
optionally	Clarify whether this means a developer choice, a system choice, or a user choice.
reasonable, when necessary, where appropriate, if possible	Explain how the developer or the user can make this judgment.
robust	Define how the system is to handle exceptions and respond to unexpected operating conditions.
seamless, transparent, graceful	Translate the user's expectations into specific observable product characteristics.
several, some, many, few	State how many, or bound a range.
shouldn't, won't	Try to state requirements as positives, describing what the system *will* do.
support	Define exactly what functions the system will perform that constitute "supporting" some capability.
user-friendly, simple, easy	Describe system characteristics that will satisfy the customer's usage needs and usability expectations.