Requirements Engineering + User Stories

Davide Falessi

Why Requirements?

- Why develop system/software/product?
    - There is problem to solve or need to meet.

- A successful product is one that will solve problem or meet the need of its users.

- Requires knowing what is the problem need to be solved (and what is needed to solve it).

What is a Requirement?

- A requirement is an expression of a system's desired behavior in order to satisfy its objective (i.e., to address the real world problem).

- Requirements focus on the customer needs, not on the solution or implementation.

- They should specify **what** behavior, without saying **how** the behavior will be realized.

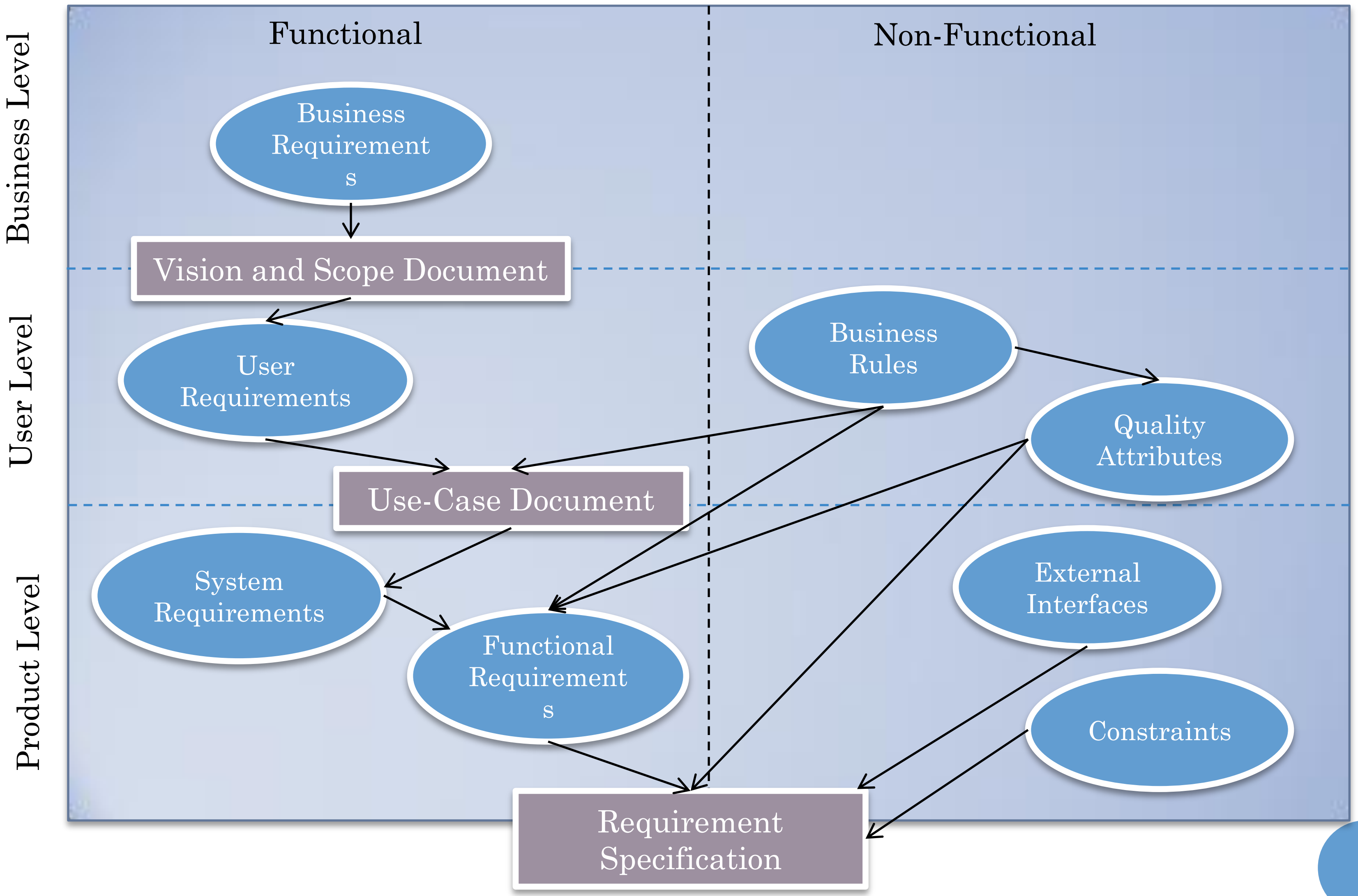
IEEE Definition of Requirement

(1) A condition or capability needed by a user to solve a problem or achieve an objective.

(2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.

(3) A documented representation of a condition or capability as in (1) or (2).

Levels of Requirements



Wiegers, Software Requirements

Importance of Requirement

- Requirements are <u>the basis</u> for system/software development.
  - They have to be discovered and understood thoroughly or the project will fail.
  - Errors occurred in requirement phase are costly to fix – especially when they propagate to later phases.
    - Requirement error found in testing costs 100 times more than a coding error found in testing.

Importance of Requirement

- Developing requirements is not trivial.
  - “The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.” (Fred Brooks, No Silver Bullet: Essence and Accidents of Software Engineering)

Difficulty in Capturing Requirement
Possible So

Stakeholders - users

<u>Communication Issues</u>
- Needs are often unclear in the beginning.
- Stakeholders may have hidden motive.
- Domain knowledge exists and they are hard to transfer.
- Hidden assumptions.
- Differentiating needs and wants.
- Communication skills may be limited.
- Expressing the understanding accurately and clearly is difficult.
- Stakeholders do not have the time to be involved.

<u>Management Issues</u>
- Requirements were originated from inappropriate stakeholders.
- Requirements evolve constantly.
- Requirements are not properly documented, tracked, and managed.

Requirement Analyst

How Do We Avoid Project Failure?

- One main factor is getting the <u>right</u> requirements <u>early</u> in the development process.
- Establish good requirements development and management practices → Requirement engineering process.

Requirement Elicitation Techniques

- Traditional techniques:
  - Reading existing documents, interviews, surveys or questionnaires.
- Collaborative techniques:
  - Group techniques (focus groups, brainstorming), prototyping, JAD/RAD workshops.
- Cognitive techniques:
  - Protocol/think aloud analysis, knowledge elicitation techniques (card sorting, laddering)
- Contextual approach:
  - Participant observation.

Requirement V&V

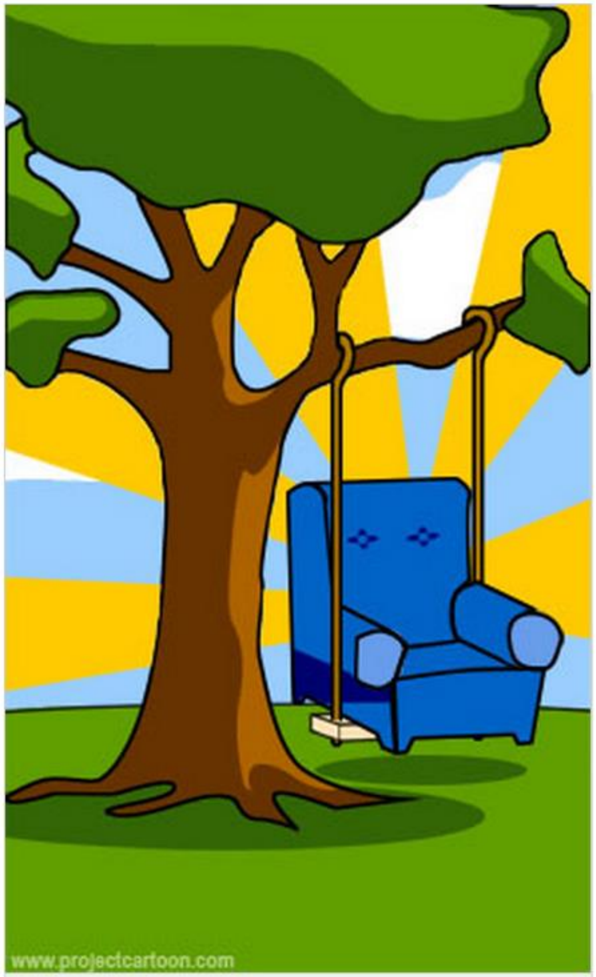
- In requirements validation, we check that our requirements definition accurately reflects the customer's needs.
- In verification, we check that one document or artifact conforms to another.
  - Check requirement document against guidelines, standard, convention, etc.
  - Check requirement specification against requirement definition (e.g. use cases).

Characteristics of a Good Requirement

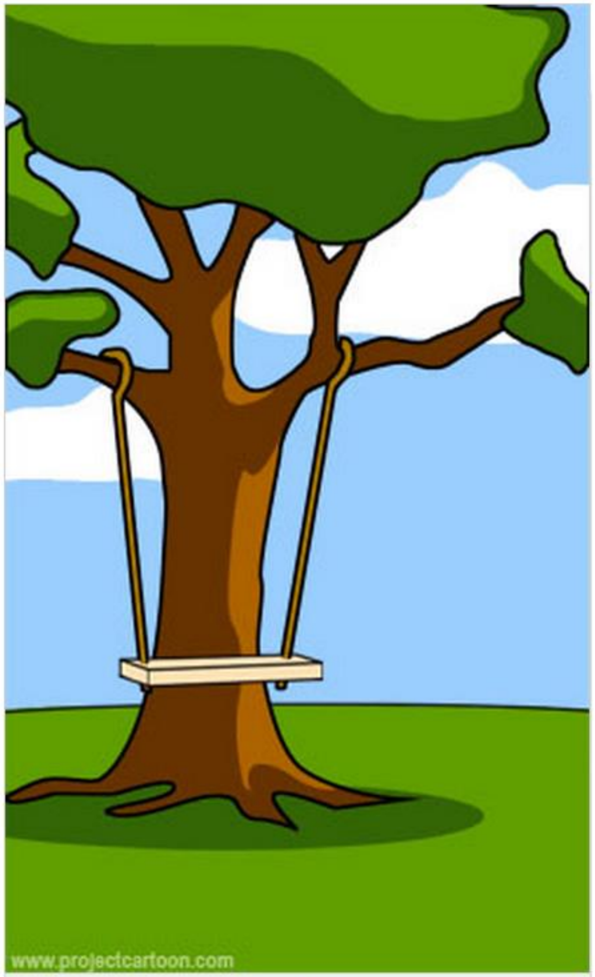
Necessary	Needed for the system to meet real needs.
Feasible	It is doable and can be accomplished within available cost and schedule.
Correct	Facts related to the requirements are accurate and it is technically and legally possible.
Concise	Requirement is stated simply.
Unambiguous	Requirement can only be interpreted in one way.
Complete	All conditions under which the requirement applies are stated, and the requirement expresses a whole idea or statement.
Consistent	Requirement is not in conflict with other requirement.
Verifiable/testable	Requirement can measured and quantified.
Traceable	Requirement can be traced to its source (originating stakeholders) and can be tracked throughout the system.
Implementation-free	Requirement does not contain a specific implementation solution.
Non-redundant	Requirement is not a duplicate of another requirement.
Stated using a standard construct	Requirement is stated as an imperative using the word shall and not compounded.
Avoid escape clause	Avoid the words *usually, generally, often, normally, typically*.

Suggested question

Can I test its implementation?



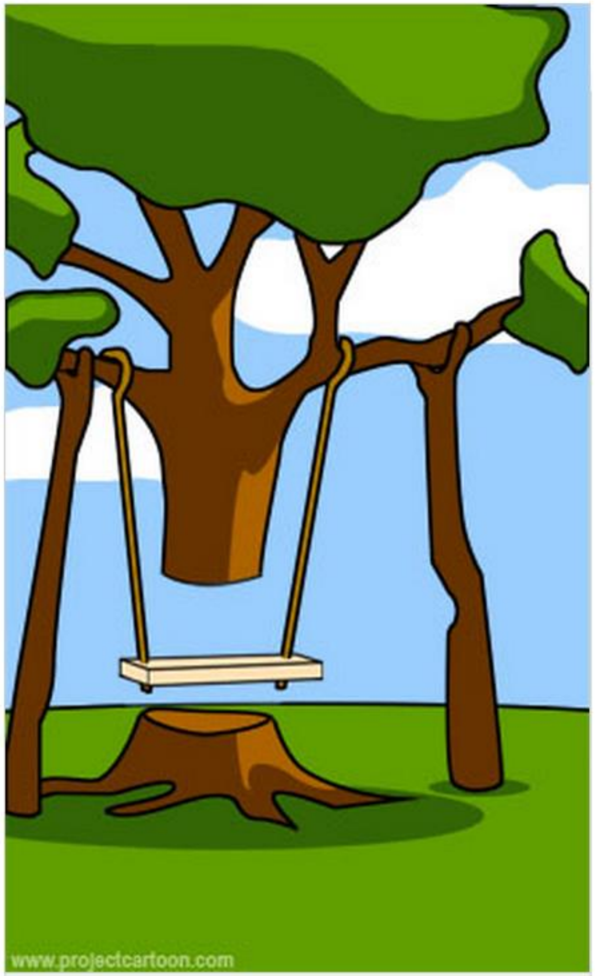
What the customer described



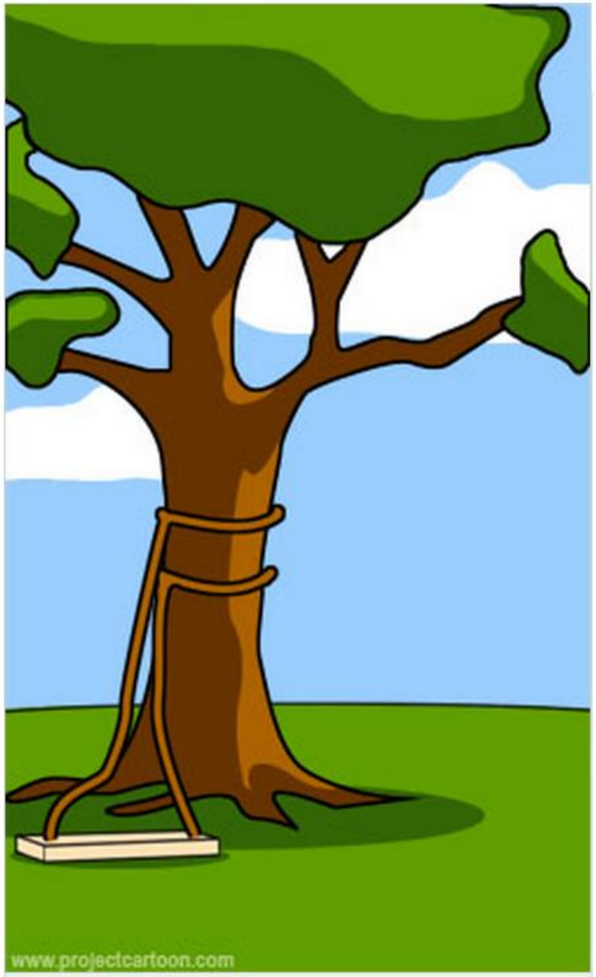
What the analyst specified

www.projectcartoon.com

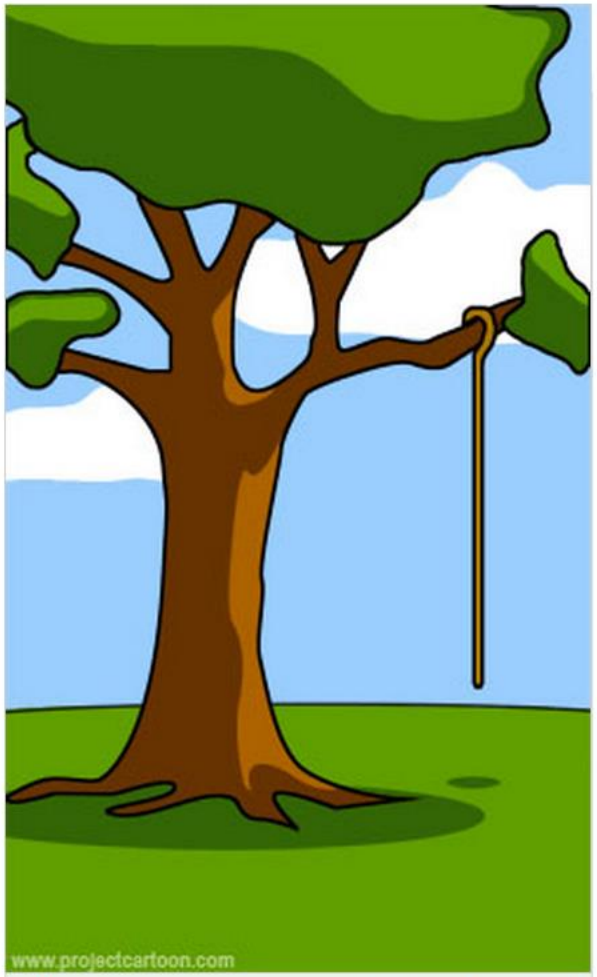
What the architects designed



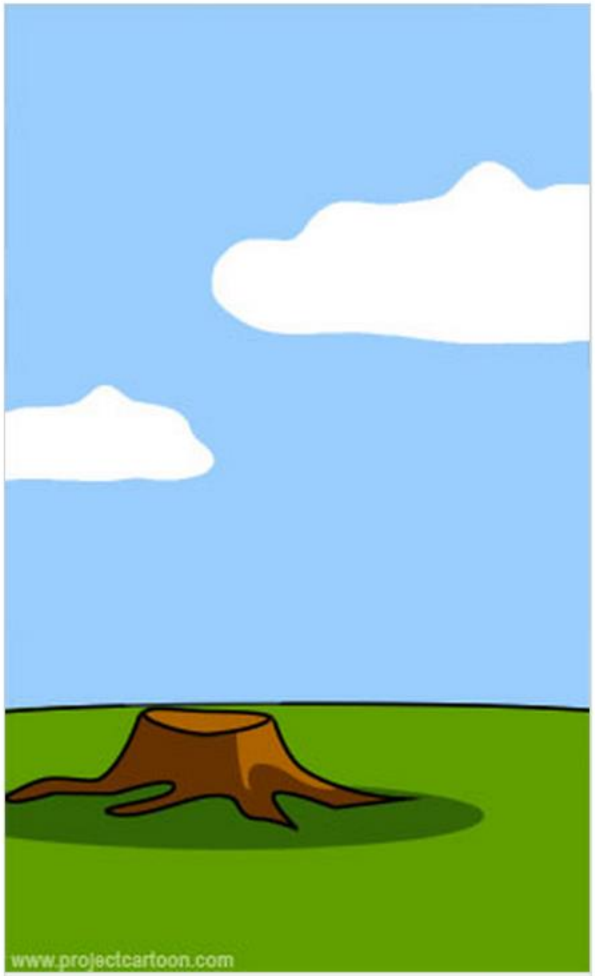
What the programmers implemented



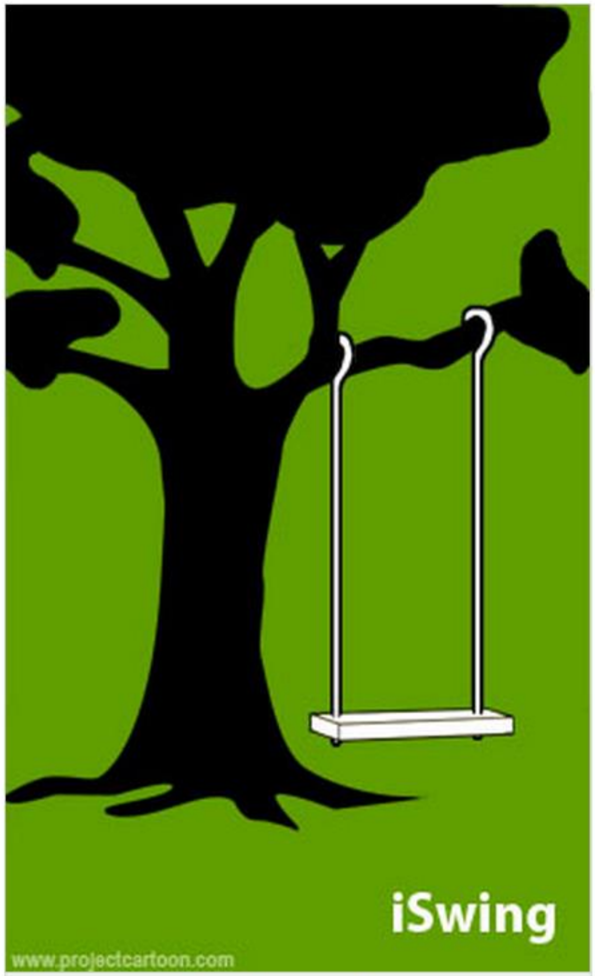
What the user manual described



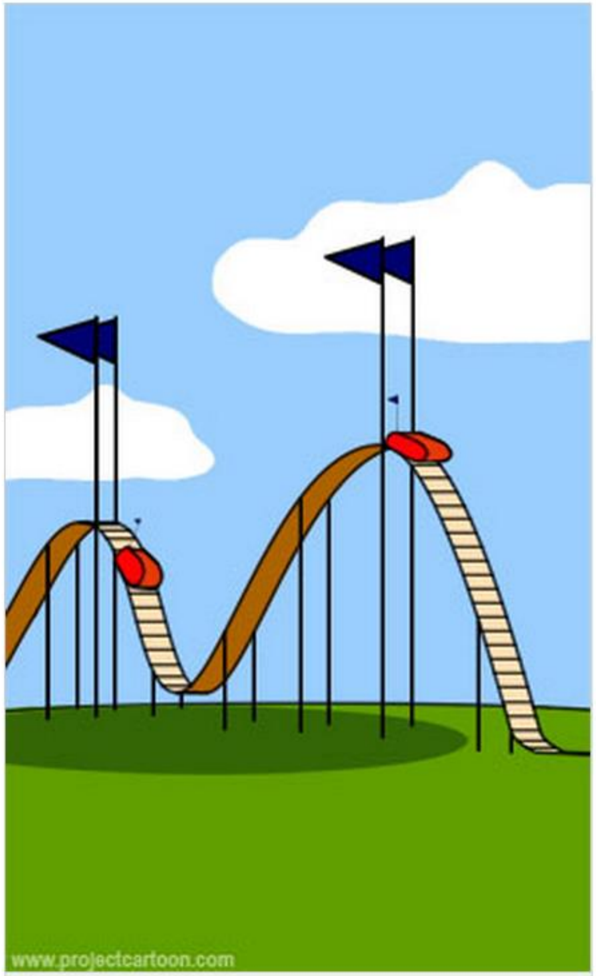
What management budgeted for



What the documentation described



How marketing advertised it



How the customer was billed

www.projectcartoon.com

What the customer really needed

User Stories

- A user story template (CARD) describes the (values of) functionality from a user's perspective.

As a <<user role>>,

I want to <<do something with the application>>,

So that <<I get specific benefits>>.

Examples of User Stories

- As a power user, I want to specify files or folders to backup based on file size, date created and date modified, so that I can be sure that my important data is saved.
- As a user, I want to indicate folders not to backup, so that my backup drive isn't filled up with things I don't need saved.