1 Nonfunctional Requirements

Nonfunctional Requirements has many factors/attributes:

1. define attributes such as security, reliability, performance, maintainability, scalability, usability, etc. a/k/a system qualities
2. functional requirements defined in user stories, etc., represent work that adds value to the user. NFR function as constraints on building decisions.
3. (a) functionality,
   (b) reliability,
   (c) performance,
   (d) supportability,
   (e) compilance,
   (f) security,
   (g) resiliance,
   (h) private,
   (i) accessibility,
   (j) regulatory standards,
4. NFR show up in multiple backlogs, but not as backlog items.
5. NFR constrain backlog items.
6. the quality assurance may have qualities test to see if system satisfies non-functional requirements.
7. Development impact: specifying 99.999 availability may be significantly more expensive than 99.9 availability.
8. NFR need to be analzized just like functional requirements.
9. compliance NFR: the NFR requirement items may be traceble to regulatory rule.
10. Set based-NFR provides alternative NFRs: e.g. uptime must be XYZ, or backup server must be up-and-running within XYZ time.
11. This allows for flexibility in design; and potentially lower cost (as the cheaper option may be chosen).
12. How to specify NFRs:
    (a) name it,

(b) scale it (units of measurement)

(c) how to measure

(d) target: what's a success

(e) constraint: failure level to avoid

(f) baseline: current level

13. requirements should specify bounds: shouldn't use: "be better"... e.g.: "airplane system should be more robust than a word processor"...

14. independent: NFRs should be independent of each other (their own dimension).

15. negotiable: should be adjustable with business negotiation

16. testable: should be testable.

17. Operational Hardening: Ensuring that suff runs, no matter what. e.g. turing machine with an initite tape... means even huge jobs should run on 1 node with infinite disk.