In software engineering, non-functional requirements (NFRs) specify criteria that describe how a system should behave, rather than what the system should do. They focus on aspects such as performance, usability, reliability, security, and maintainability. Non-functional requirements are essential for ensuring that the system operates effectively and meets the needs of its users. Some common categories of non-functional requirements include:

1. Performance: This includes metrics related to response time, throughput, scalability, and resource usage. For example, the system may be required to handle a certain number of concurrent users without degradation in performance.

2. Usability: Usability requirements describe how user-friendly the system should be. This could include factors such as interface design, accessibility, and learnability.

3. Reliability: Reliability requirements pertain to the system's ability to perform its functions consistently and accurately over time. This may involve metrics such as mean time between failures (MTBF) or mean time to repair (MTTR).

4. Security: Security requirements define measures to protect the system and its data from unauthorized access, breaches, or other security threats. This might include authentication, encryption, and access control mechanisms.

5. Maintainability: Maintainability requirements focus on how easily the system can be modified, updated, or repaired. This could involve factors such as modularity, documentation, and the use of standard coding practices.

6. Scalability: Scalability requirements address the system's ability to handle increasing loads or growing data volumes without significant performance degradation. This might involve horizontal or vertical scaling strategies.

7. Availability: Availability requirements specify the amount of time the system should be operational and accessible to users. This could include uptime targets or requirements for failover and redundancy.

8. Interoperability: Interoperability requirements define how the system should interact with other systems or components. This might involve adherence to industry standards or compatibility with specific software or hardware.

9. Compliance: Compliance requirements ensure that the system meets legal, regulatory, or industry standards. This could include data privacy regulations, industry-specific guidelines, or international standards.

10. Resilience: Resilience requirements address the system's ability to recover from failures or disruptions. This might involve strategies such as fault tolerance, disaster recovery, or graceful degradation.

Non-functional requirements are often just as critical as functional requirements in determining the success of a software system. They help ensure that the system not only meets the functional needs of its users but also delivers a satisfactory overall experience in terms of performance, security, usability, and other important qualities.