**Was sind Anforderungen**

**Anforderungen sind die Aufgaben, die im Programm vorhanden sein müssen und die von dem Kunden definiert worden.**

Das Lastenheft dient dazu, die Anforderungen an das zu entwickelnde System verbindlich zu beschreiben. Innerhalb des Lastenhefts werden die relevanten Anforderungen durch den Auftraggeber ermittelt und dokumentiert. Zu den Anforderungen gehören sowohl funktionale als auch nicht funktionale Anforderungen an das System. Weitere Bestandteile sollten nach Möglichkeit eine Skizze des Gesamtsystems, Richtlinien für Technologieentscheidungen und die Formulierung von Vorgaben für die vertragliche Gestaltung, wie Lieferbedingungen und Abnahmekriterien sein. Nach Möglichkeit sollten die Anforderungen so aufbereitet sein, dass eine Nachverfolgbarkeit (Traceability) gegeben ist. Bezüglich der technischen Umsetzungen sollten keine Detailvorgaben gemacht werden, um den Prozess der Lösungssuche (durch den Auftraggeber) nicht im Vorfeld bereits zu stark einzuschränken und zu strukturieren.

**Functional & non functional requirements**

**Funktionale, nicht funktionale Anforderungen**

Im Requirements Engineering unterscheidet man zwischen funktionalen und nicht funktionalen Anforderungen.

Funktionale Anforderungen beschreiben gewünschte Funktionalitäten (was soll das System tun/können) eines Systems bzw. Produkts, dessen Daten oder Verhalten.

Nichtfunktionale Anforderungen sind Anforderungen, an die "Qualität" in welcher die [geforderte Funktionalität](http://www.anforderungsmanagement.ch/glossar/6.html) zu erbringen ist.

Qualität im vorgenannten Sinn meint beispielsweise

* wie die Funktionalität ausgeführt werden soll (z.B. Reaktionszeit)
* Bedingungen unter denen die Funktionalität ausgeführt wird (z.B. 7x24 Std.)

Die Unterscheidung zwischen funktionalen und nicht funktionalen Anforderungen ist nicht das Wesentliche - wesentlich ist, dass beide Arten im Projekt vollständig berücksichtigt werden.

**Unterschiede**

Allerdings lassen sich Gründe definieren, weshalb die Unterscheidung zwischen funktionalen und nicht funktionalen Anforderungen wesentlich sein kann:

**Funktionale Anforderungen** unterscheiden sich von Produkt zu Produkt; d.h. wenn eine Bank verschiedene Applikationen baut, werden sich die Funktionen in diesen Applikation sehr wahrscheinlich unterscheiden. **Nicht funktionale Anforderungen** hingegen können für viele Produkte derselben Unternehmung vorgegeben/vorgeschlagen oder sogar genormt werden (z.B. einheitliches Erscheidungsbild und damit "ähnliche" Usability; Vorgaben von Betriebsmodi welche von der Organisation unterstützt werden, etc.)

Ein zweiter Grund liegt darin, dass sich die **nicht funktionalen Anforderungen** häufig auf die gesamte Architektur, das gesamte Design des Produktes auswirken, während die **funktionalen Anforderungen** normalerweise nur "lokale" Auswirkungen haben; d.h. wenn an den nicht funktionalen Anforderungen "geschraubt" wird, hat dies normalerweise viel weitergehende Konsequenzen als bei den rein funktionalen Anforderungen.

Ein dritter Grund liegt darin, dass **nicht funktionale Anforderungen** häufig erst/nur dann als Anforderungen erkannt werden, wenn in der Vergangenheit schlechte Erfahrungen mit diesem Thema gemacht worden sind. Oder würden Sie eine Anforderung an die Performance ihres neuen PC definieren, wenn Sie bisher nie Probleme mit der Performance gehabt hätten?

**Specifications**

Specifications sind Lösungen der Anforderungen.

Wie wird eine bestimmte Anforderung des Kunden im Programm umgesetzt, um diese zu realisieren.

Die Specification beschreibt die Lösung, die verwendet wurde, um die Anforderungen umzusetzen.

## **Stakeholder**

Stakeholder sind folgende Personen, die den Auftrag zum Erstellen der Software stellen oder Teilhaber dieser Software sind. Das sind die Kunden und diese werden die Anforderungen des Programms auflisten.

Mit Stakeholder (auf Deutsch: Anspruchsgruppen) werden alle Personen, Gruppen oder Institutionen bezeichnet, die von den Aktivitäten eines Unternehmens direkt oder indirekt betroffen sind oder die ein Interesse an diesen Aktivitäten haben. Die Stakeholder versuchen, auf das Unternehmen Einfluss zu nehmen.

**Userstory**

User Story ist ein Werkzeug, um gewünschte Funktionalitäten eines Systems aus Sicht des Anwenders zu beschreiben. Dabei bietet eine User Story vor allem drei Vorteile:

* sie ist leicht zu verstehen und vermittelt die Wünsche der Anwender
* sie ist schnell erstellt und erleichtert die Schätzung des Aufwands zur Realisierung
* sie lässt sich schrittweise detaillieren und unterstützt so die iterative Entwicklung

Als Konzept geht die User Story auf Extreme Programming (XP) zurück. Extreme Programming ist ein agiles Entwicklungsmodell

## **Wie man eine User story erstellt:**

Es geht vor allem um

* WER,
* WAS und
* WARUM,

also wer möchte was von einem System, um welchen Nutzen davon zu haben. Wie die gewünschte Funktionalität später umgesetzt wird, ist für eine User Story unerheblich.

## **User Story Beispiele**

Hier finden Sie einige Beispiele für User Storys:

* Als Filmliebhaber möchte ich über neue Filme informiert werden, um zu wissen, welche Filme als nächstes im Kino laufen.
* Als Filmliebhaber möchte ich einmal pro Woche einen Newsletter erhalten, um zu wissen, welche Filme als nächstes im Kino laufen.
* Als Filmliebhaber möchte ich einmal pro Woche per Mail über neue Science Fiction Filme informiert werden, die im Colosseum Kino laufen, um mir für entsprechende Filme in diesem Kino Tickets online buchen zu können.