Quality Knowledge Letter
20.12.2004

Ausgabe 2004 / 5
Erscheinungsart: ca. 4 x jährlich in elektronischer Form

# Nicht-funktionale Anforderungen

weiters in dieser Ausgabe ...



## Kurzdefinition / Glossar ...

Eine funktionale Anforderung ist eine Anforderung an das System, die man als Ergebnis beobachten kann, wenn das System in Betrieb ist.
Eine **NICHT-Funktionale Anforderung** (NFA) ist eine Beschreibung einer Beschaffenheit oder einer Charakteristik, die ein System aufweisen muss oder eine Rahmenbedingung die eingehalten werden muss, die den Freiheitsgrad bei der Konstruktion des Systems (also die Umsetzung der funktionalen Anforderungen) einschränkt.

## NICHT-funktionale Anforderungen - Ein wesentlicher Teil der Spezifikation!

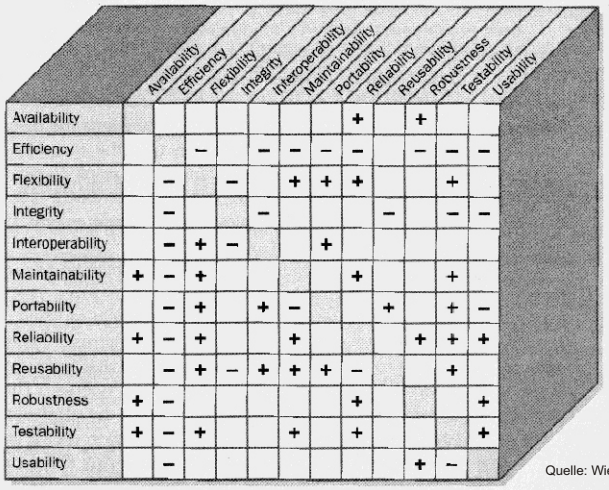
Ein System muss nicht nur die geforderte fachliche Funktionalität abbilden, sondern auch entscheidende nicht funktionale Anforderungen erfüllen.

Nicht-funktionale Anforderungen können je nach Kritikalität und Einsatzgebiet des Systems weitreichende Folgen haben.

Außerdem haben sie weitreichenden Einfluss auf Designentscheidungen. Sehr viele Designentscheidungen werden maßgeblich durch nicht-funktionale Anforderungen gesteuert und nicht dem Zufall überlassen.

Was jedoch vielfach übersehen wird ist, dass auch nicht-funktionale Anforderungen abnahme-relevant sind und daher auch Abnahmekriterien benötigen. Die bei funktionalen Anforderungen geforderte Testbarkeit gilt hier genauso!

Ebenso ist zu berücksichtigen, dass eine gegenseitige Abhängigkeit und Beeinflussung zwischen vielen nicht-funktionalen Kriterien besteht (siehe Grafik):




Quelle: Wiegers, 2003

Es gibt an jedes System theoretisch unendlich viele nicht-funktionale Anforderungen. In der Praxis haben sich verschiedene Gliederungs-Systematiken etabliert, die versuchen, die wesentlichen nicht-funktionalen Kriterien übersichtlich zu strukturieren. Das bekannteste Modell dabei ist die ISO 9126. Daneben gibt es in der Literatur noch viele weitere Modelle, auf die hier jedoch aus Platzgründen nicht eingegangen wird. Beispielhaft ist im Artikel auf Seite 2 ein Gliederungsschema angeführt.

### Der vernachlässigte Teil

Wenn man verschiedene Pflichtenhefte durchsieht (ich sammle seit Jahren Pflichtenhefte von öffentlichen Ausschreibungen), dann fällt auf, dass die NFA's sehr häufig fehlen oder völlig unzureichend spezifiziert sind.
Einerseits (bei Fehlen) ist dabei absehbar, dass es sehr wahrscheinlich in der Inbetriebnahme- und Betriebs-Phase zu unangenehmen Diskussionen kommen wird, wenn plötzlich durch den Auftraggeber neben der 'üblichen' Änderung von Anforderungen während des Projekts auch noch nicht-funktionale Anforderungen und Wünsche geäußert werden, die bislang gar nicht oder nur als implizite Erwartungen in seinem Kopf existierten.
Andererseits (wenn sie spezifiziert wurden) sind diese NFA's meist so unklar definiert, dass sie von jedem beteiligten Partner unterschiedlich interpretiert werden (können).
Problematisch ist dabei, dass sich der Auftraggeber meist bis zum Schluss des Projekts 'in Sicherheit' wiegt, da er ja auch NFA's 'spezifiziert' hat.
Wenn NFA's spezifiziert werden, sollte daher besonders darauf geachtet werden, dass nur Anforderungen spezifiziert werden, die auch im Rahmen des Projekts bzw. bei der Abnahme klar überprüft bzw. gemessen werden können.

**Dipl.-Ing. Johannes Bergsmann**
allgemein gerichtlich beeideter und zertifizierter Sachverständiger für Informatik

Der Quality-Knowledgeletter ist ein periodisches Informationsmedium von Software Quality Lab und dessen Partnern mit den Schwerpunkten IT-Qualitätsmanagement, Projekt- und Prozess-Management.
Inhalt: fachliche Beiträge und Schwerpunktthemen, Vorstellung neuer Produkte und Leistungen, neue wissenschaftliche Erkenntnisse, ...
Aktuelle Fach- und Forschungsbeiträge sind willkommen. Einsendungen an info@software-quality-lab.at .
**Weitere Infos zu diesem und anderen Themen finden Sie auf http://www.software-quality-lab.at .**

Quality Knowledge Letter
20.12.2004

# Nicht-Funktionale Anforderungen

Für nichtfunktionale Anforderungen gibt es zahlreiche Quellen (siehe auch letzte Seite dieser Ausgabe), die großteils ähnliche Inhalte aufweisen, jedoch teilweise sehr unterschiedlich strukturiert sind.

Das nachfolgend angeführte Schema orientiert sich grundsätzlich an der Struktur der ISO 9126 und ergänzt diese Struktur dort wo es sinnvoll ist durch Elemente aus dem Modell FURPS von HP.

## ➲ Usability / Benutzbarkeit

Beschreibt Kriterien, die für die Benutzung und die individuelle Beurteilung der Benutzung durch eine bestimmte Benutzergruppe erforderlich sind.

Teilaspekte sind:

> Understandability / Verständlichkeit

Beschreibt die Möglichkeiten des Benutzers, das System-Konzept zu verstehen und zu erkennen, ob das System für seine Anforderungen passt und wie er mit diesem System umgehen soll.

> Ease of Learning /Erlernbarkeit

Aspekte für den Benutzer, um das System zu erlernen (z.B. Eingabe/Ausgabe, Bedienung, ...). Diese Anforderung sollen den Designern zeigen, wie die Anwender das System erlernen.

> Operability / Bedienbarkeit

Anforderungen an das Handling / die Steuerung der Anwendung. Hier werden auch ergonomische Aspekte berücksichtigt.

> Attractiveness (Look&Feel)

Aspekte, welche die Attraktivität für den Benutzer steigern. Die Look&Feel-Anforderungen spezifizieren die Vorstellung des Kunden vom Erscheinungsbild des Systems.

> Accessibility / Zugänglichkeit

Hier werden Aspekte beschrieben, die die Möglichkeiten für den Benutzer beschreiben, um einfachen Zugang zum System zu erhalten. Hier werden z.B. Kriterien beschrieben für Benutzer mit Einschränkungen (körperliche Behinderungen, ...).

## ➲ Reliability / Zuverlässigkeit

Kriterien, die die Fähigkeit des Systems spezifizieren, das Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum aufrecht zu halten.

> Maturity / Reife

Definiert die Versagenshäufigkeit durch Fehlerzustände.

> Fault-Tolerance / Fehlertoleranz

Definiert die Fähigkeit, ein festgelegtes Leistungs-Niveau bei Software-Fehlern oder Nichteinhaltung der spezifizierten Schnittstelle zu erhalten.

> Recoverability /Wiederherstellbarkeit

Definiert die Fähigkeit des Systems, bei einem Versagen das Leistungsniveau wieder herzustellen und die direkt betroffenen Daten wieder zu gewinnen.

## ➲ Efficiency /Effizienz

Beschreibt die Fähigkeit des Systems, ein angemessenes Leistungsniveau relativ zu den dafür eingestezten Betriebsmitteln bereit zu stellen.

> Time Behaviour / Zeitverhalten

Beschreibt Anforderungen, um eine angemessene Antwort- und Verarbeitungszeit sowie Durchsatz zur Verfügung zu stellen.

> Resource utilisation /Verbrauchsverhalten

Kriterein für die Anzahl und Dauer der für die Funktionserfüllung benötigten Betriebsmittel.

## ➲ Performance / Leistungsverhalten

Beschreibt alle nicht direkt auf funktionale Bereiche bezogene Eigenschaften wie z.B. Genauigkeiten von Ergebnissen und Datenmengen mit dem das System umgehen können soll.

## ➲ Maintainability / Wartbarkeit

Beschreibt die Möglichkeiten, das System zu modifizieren (Korrekturen, Verbesserungen, Anpassungen, ...). Dabei sind alle nicht direkt funktionale Kritieren wesentlich wie z.B. benötigte Zeit für Änderungen, geplante Release-Zyklen, den benötigten Support-Level, Fähigkeiten der Programmiersprachen, ...

> Analyzability / Analysierbarkeit

Beschreibt die Anforderungen, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungs-bedürftige Teile zu bestimmen.

> Changeability / Modifizierbarkeit

Beschreibt die Anforderungen, um eine möglichst einfache Umsetzung von spezifizierten Änderungen zu ermöglichen.

> Stability / Stabilität

Definiert Anforderungen, um die Wahrscheinlichkeit des Auftretens von unerwarteten Effekten bei vorgenommenen Änderungen möglichst gering zu halten.

> Testability / Prüfbarkeit

Anforderungen bezüglich der Prüfung des geänderten Systems und bezüglich des Aufwands für die Prüfung des geänderten Systems.

## ➲ Reusability / Wiederverwendbarkeit

Beschreibt die Möglichkeiten, das System oder auch einzelne Systemkomponenten in anderen Entwicklungs-Projekten oder auch als COTS-Produkte (Commercial-of-the-shelf = Standard- / 'Von-der-Stange'-Produkte) wieder zu verwenden.

Fortsetzung auf nächster Seite >>>

# Vollständiger Knowledge Letter Zugang

Wir freuen uns, dass Sie an diesem Thema Interesse haben und den Knowledge Letter von Software Quality Lab bis hierher gelesen haben.

**Dieser Knowledge Letter ist eine Vorschau (gekürzte Version des gesamten Artikels).**

Wenn Sie den ungekürzten Knowledge Letter lesen möchten, registrieren Sie sich bitte unter
**http://www.software-quality-lab.com/download/knowledge-letter/anfrage-knowledge-letter/**

Sie erhalten nach der Registrierung vollen Zugang zu allen bisherigen Knowledge Lettern von Software Quality Lab und erhalten automatisch künftige Knowledge Letter per E-Mail.