

## Ausgabe 2004 / 5

Erscheinungsart: ca. 4 x jährlich in elektronischer Form

# Nicht-funktionale Anforderungen

weitere in dieser Ausgabe ...

- ⇒ NICHT-funktionale Anforderungen - Ein Überblick
- ⇒ Literatur, Links, Zitate

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):

	Availability	Efficiency	Flexibility	Integrity	Interoperability	Maintainability	Portability	Reliability	Reusability	Robustness	Testability	Usability
Availability												
Efficiency			-	-	-	-	-	-	-	-	-	-
Flexibility		-	-		+	+	+					+
Integrity		-		-					-	-	-	
Interoperability		-	+	-			+					
Maintainability		+	-	+				+				+
Portability		-	+		+	-			+			+
Reliability		+	-	+		+				+	+	+
Reusability		-	+	-	+	+	+	-				+
Robustness		+	-					+				+
Testability		+	-	+		+	+					+
Usability			-							+		-

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 Bergmann

allgemein gerichtlich beideter 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](mailto:info@software-quality-lab.at).

Weitere Infos zu diesem und anderen Themen finden Sie auf <http://www.software-quality-lab.at>.

## 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 eingestetzten Betriebsmitteln bereit zu stellen.

#### > Time Behaviour / Zeitverhalten

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

#### > Resource utilisation / Verbrauchsverhalten

Kriterien 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 Kriterien 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 >>>

Fortsetzung - NICHT-funktionale Anforderungen >>>

### ↻ **Portability / Übertragbarkeit**

Beschreibt die Eignung des Systems, von einer Umgebung (organisatorische, Hardware- oder Software-technische, ...) in eine andere übertragen zu werden.

#### > Adaptability / Anpassbarkeit

Beschreibt die Möglichkeit, das System an verschiedene festgelegte Umgebungen anzupassen, wenn nur Schritte unternommen oder Mittel eingesetzt werden, die für diesen Zweck für dieses System schon vorgesehen sind.

#### > Installability / Installierbarkeit

Anforderungen, die für das Installieren des Systems in einer anderen Umgebung notwendig sind.

#### > Co-Existence / Koexistenz

Beschreibt die Möglichkeit, das System gemeinsam mit anderen unabhängigen Systemen in einer gemeinsamen Umgebung eventuell auch unter Verwendung von gemeinsamen Ressourcen zu betreiben.

#### > Replaceability / Austauschbarkeit

Beschreibt die Möglichkeit, das System anstelle eines spezifizierten anderen Systems in der Umgebung dieses Systems mit demselben Einsatzzweck zu verwenden.

### ↻ **Scalability / Skalierbarkeit**

Beschreibt Anforderungen des Systems an den Umgang mit steigenden Nutzerzahlen oder Datenaufkommen (z.B. mengenmäßige Ausweitung des Einsatzbereichs von einer Organisationseinheit auf alle Organisationsbereiche).

### ↻ **Security / Sicherheit (des Systems)**

Beschreibt Anforderungen des Systems an die Sicherheit (z.B. unberechtigter Zugriff auf Programme und Daten).

### ↻ **Safety / Sicherheit (der Anwender)**

Beschreibt Anforderungen an das System, um Risiken, die durch das System gegeben sind und die für die Anwender des Systems ein personenbezogenes Gefahrenpotential (z.B. körperliche Verletzungsgefahr) darstellen, möglichst gering zu halten.

### ↻ **Documentation / Dokumentation**

Definiert Kriterien (Art, Umfang, Beschaffenheit, strukturelle und inhaltliche Vorgaben) für die Dokumentation des Systems sowohl technische als auch Anwender-Dokumentation.

Es gibt noch weitere, hier nicht näher beschriebene Kriterien, die bei Bedarf als eigener Punkt in die Spezifikation aufgenommen werden können wie z.B. Konfigurierbarkeit, Konsistenz, Kompatibilität, ...

## Es war einmal ...

Ein großes Unternehmen entwarf einst ein ausgeklügeltes neues grafisches e-Business-System. Die Kunden sollten im Web-Shop Produkte kaufen und verschiedenste Services abrufen können.

Die Grafik der Benutzeroberfläche war atemberaubend, der Aufbau war genial.

Das Benutzerinterface arbeitete über das LAN während der Entwicklungs-Phase sehr gut.

Unglücklicherweise verwendeten viele Kunden noch eine langsame Modem-Leitung und mussten dann nach Fertigstellung des Systems eine ärgerlich langsame Download-Rate der riesigen Grafik-Dateien feststellen.

Beta-Tester verloren schon nach kurzer Zeit aufgrund der miserablen Performance das Interesse an dem System.

In Ihrem Enthusiasmus über das faszinierende grafische Benutzer-Interface vergaßen die Entwickler, die Rahmenbedingungen für den Betrieb vorab zu untersuchen und festzulegen.

Der gesamte Ansatz wurde nach der Fertigstellung wieder verworfen und musste neu entwickelt werden.

*Beispiel für eine sehr teure Lektion über die Wichtigkeit einer frühzeitigen Ermittlung und Festlegung von Qualitätskriterien in SW-Projekten.*

## Leistungen ...

Im Bereich Anforderungsmanagement und Spezifikation unterstützen wir Sie durch folgende Dienstleistungen:

- ⇒ Erstellung von Spezifikationen (speziell auch für Ausschreibungen)
- ⇒ Requirements-Engineering Workshops gemeinsam mit Auftraggeber (und ev. auch Auftragnehmer)
- ⇒ Prüfen von Spezifikationen vor Ausschreibungen
- ⇒ Unterstützung bei der Erstellung Vorlagen und Checklisten für Spezifikationen, ...
- ⇒ Ausbildung von Requirements-Managern
- ⇒ begleitendes firmeninternes Coaching nach Bedarf

Weitere Infos und Leistungsbereiche finden Sie auf unserer Web-Site [www.software-quality-lab.at](http://www.software-quality-lab.at).

## Zitate ...

- ⇒ Wenn der Zug in die falsche Richtung fährt, sind alle Stationen falsch!  
*Franz Josef Strauß*
- ⇒ Nicht zum Zwecke der gegenseitigen Täuschung wurde die Sprache dem Menschen gegeben, sondern damit er seine Gedanken anderen mitteilen möge.  
*Augustinus*

### Impressum:

Herausgeber und für den Inhalt verantwortlich:

Die Beiträge wurden sorgfältig ausgewählt und bearbeitet. Für Druckfehler und Irrtümer wird nicht gehaftet. Die Rechte von zitierten Beiträgen und Gastbeiträgen liegen bei den jeweiligen Autoren.

### Software Quality Lab

Fliederstrasse 8  
A-4222 Langenstein

[www.software-quality-lab.at](http://www.software-quality-lab.at)

[info@software-quality-lab.at](mailto:info@software-quality-lab.at)

Tel.: +43-(0)664-16 20 220, Fax: +43-(0)7237-4941