Qualitätssicherung

<Kunde>

<Projektname>

Autor: Mohammad Abuosba

Letzte Änderung: 17. Dezember 2015

Dateiname: muster\_qualitätssicherung.docx

Version: 0.3

***Inhaltsverzeichnis***

[1 Einleitung 4](#_Toc375215632)

[2 Qualitätsplan 6](#_Toc375215633)

[3 Testdurchführung – ein Prozessüberblick 10](#_Toc375215634)

[4 Testfälle 12](#_Toc375215635)

[4.1 Testfallbeschreibung 12](#_Toc375215636)

[4.2 Beispiel eines Testfalls: Import einer CSV-Datei In das GUI Tool 13](#_Toc375215637)

[4.3 Beispiel eines Testfalls: M/W Erkennung durch den Namen 14](#_Toc375215638)

[4.4 Beispiel eines Testfalls: GUI-Kommunikation 15](#_Toc375215639)

[5 Testprotokoll 16](#_Toc375215640)

[6 Anhang 17](#_Toc375215641)

[6.1 Fehlerkategorien 17](#_Toc375215642)

[6.2 Q-Kriterien ISO 9126 17](#_Toc375215643)

[6.3 Q-Kriterien für Dokumente 19](#_Toc375215644)

***Abbildungsverzeichnis***

[Abbildung 1: Prozessmodell 4](#_Toc375215645)

[Abbildung 2: das Magische Dreieck 5](#_Toc375215646)

[Abbildung 3: Testorganisation und –ablauf 10](#_Toc375215647)

***Copyright***

© Mohammad Abuosba

Die Weitergabe, Vervielfältigung oder anderweitige Nutzung dieses Dokumentes oder Teile davon ist unabhängig vom Zweck oder in welcher Form untersagt, es sei denn, die Rechteinhaber/In hat ihre ausdrückliche schriftliche Genehmigung erteilt.

***Version Historie***

|  |  |  |  |
| --- | --- | --- | --- |
| *Version:* | *Datum:* | *Verantwortlich* | *Änderung* |
| 0.1 | 11.11.2013 | Mohammad Abuosba | Initiale Dokumenterstellung |
| 0.2 | 15.12.2013 | Mohammad Abuosba | Erweiterungen |
| 0.3 | 19.12.2013 | Mohammad Abuosba | Finalisierung |
| 0.4 |  |  |  |
| 0.5 |  |  |  |
| 1.0 |  |  |  |
| 1.1 |  |  |  |
| 1.2 |  |  |  |

***Vorhandene Dokumente***

Alle für die vorliegende Spezifikation ergänzenden Unterlagen müssen hier aufgeführt werden

|  |  |  |
| --- | --- | --- |
| Dokument | Autor | Datum |
| Fachliche Anforderungen AX.doc | Denise Müller | 19.09.2013 |
| Pflichtenheft xy.doc | Manfred Meyer | 30.09.2013 |
| Technische Spezifikation ABC.docx | Anton Schulze | 11.12.2013 |
|  |  |  |
|  |  |  |

# Einleitung

Qualitätssicherung ist ein Teilprozess innerhalb des Prozessmodells und gehört zu den Supportprozessen.



Abbildung 1: Prozessmodell

Der Begriff Qualität ist abgeleitet aus dem lateinischen Wort 'qualitas': Beschaffenheit, Eigenschaft oder Eignung eines Gutes für bestimmte Verwendungszwecke (Zweckeignung).

Qualität wird definiert als die Gesamtheit von Merkmalen einer Einheit bezüglich ihrer Eignung, festgelegte und vorausgesetzte Erfordernisse zu erfüllen.

Software-Produkt-Qualität wird definiert als die Gesamtheit von Eigenschaften eines Software-Produkts, die sich auf die Eignung zur Erfüllung gegebener Erfordernisse beziehen.

Zweck des Prozesses Qualitätssicherung ist die Festlegung, Bereitstellung und der Betrieb eines effektiven verbindlichen Qualitätssicherungsverfahrens zur Sicherstellung der angeforderten Qualität der Arbeitsergebnisse des Software-Entwicklungs- bzw. Konstruktionsprozesses.

Die Ziele des Prozesses sind

* Optimierung der Effizienz (Kosten-Nutzen-Relation) der Projekt-Qualitätssicherung durch frühzeitige, angemessene Prüfung der Arbeitsergebnisse bzgl. der Qualitätsanforderungen und Einleitung frühzeitiger Korrekturmaßnahmen
* Optimierung der Effektivität (Wirksamkeit, Qualität der Zielerreichung) der Qualitätssicherung in Bezug auf die Anforderungen von Kunden
* Gewährleistung erfolgreicher Abnahmen in den Projekten
* Maßgeblicher Beitrag zur Erreichung der Qualitätsziele des Führungsprozesses und des Projektmanagementprozesses  
  (Hohe Kundenzufriedenheit: Reklamationen des Kunden mit rückläufiger Tendenz; Hohe Effizienz im Spannungsdreieck (Kosten-Nutzen-Optimierung) durch angepassten Testaufwand)



Abbildung 2: das Magische Dreieck

# Qualitätsplan

Unter Qualitätsplan versteht man die Dokumentation der Teststrategie. Folgende Festlegungen werden im Q-Plan dokumentiert:

* Festlegung der benötigten Teststufen
* Festlegung der benötigten Testverfahren
* Herausarbeitung eventuell vorhandener besonders markante Qualitätskriterien

Der Testplan ist Teil davon und dient der Dokumentation aller zur Überprüfung der Einhaltung der Qualitätsanforderungen notwendigen Tests.

In Abhängigkeit von Anforderungen und zu testendem Projektergebnis (Ergebnistyp) werden die notwendigen Teststufen festgelegt:

**QS-Teststufe Dokumente:** Dokumenten-Tests   
-sowohl inhaltliche als auch formale Überprüfung-

**QS-Teststufe 1:** Funktionstests   
-DV-technische Überprüfung der Lauffähigkeit des Codes bzw. Konstruktion / Überprüfung einzelner Funktionalitäten und Funktionen-

**QS-Teststufe 2:** Integrationstests   
-Überprüfung des Zusammenspiels aller für das Gesamtprodukt notwendigen Komponenten (Setup, Datenbanken, Module, ggf. Konstruktion usw.)-

**QS-Teststufe 3:** Abnahmetests   
-Abschließendes Abnahmetestat durch den Kunden-

Zu jeder definierten Teststufe müssen durchzuführende Testverfahren festgelegt werden.

Folgendes Template soll angewendet werden

Testplan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test-Objekt** | **Qualitätskriterien** | **QS-Teststufe 1 "Source Code, Komponente, Funktion"** | | | **Bemerkungen** |
|
|
| **Test-Verfahren** | **Zyklus** | **Zuständig** |
| **Dokumentation** | | | | | |
| Source code | Verständlichkeit Lesbarkeit, Funktionale Vollständigkeit  und Korrektheit | Editorial Review  Technisches Review  Gegenlesen | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied,  Anwender |  |
| Source code-Dokumentation | Verständlichkeit Lesbarkeit, Funktionale Vollständigkeit  und Korrektheit t | Editorial Review  Technisches Review  Gegenlesen | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied,  Anwender |  |
| … |  |  |  |  |  |
| **Applikation** | | | | | |
| **Funktionalitäten** | | | | | |
| Unit Test | F - Richtigkeit | Datentest, Lasttest, Stresstest, Performanztest,  Ergonomie | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| Modultest | F - Richtigkeit | Datentest, Lasttest, Stresstest, Performanztest,  Ergonomie | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| … |  |  |  |  |  |
| **nicht funktionale Eigenschaften / Anforderungen** | | | | | |
| optional | Änderbarkeit, Benutzbarkeit, Effizienz, Zuverlässigkeit | Datentest, Lasttest, Stresstest, Performanztest,  Ergonomie | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| … |  |  |  |  |  |
|  | | | | | |
| **Test-Objekt** | **Qualitätskriterien** | **QS-Teststufe 2 "Integration / Systemtest"** | | | **Bemerkung?** |
|
|
| **Test-Verfahren** | **Zyklus** | **Zuständig** |
| **Funktionalitäten** | | | | | |
| generell | F - Richtigkeit | Datentest, Lasttest, Stresstest, Performanztest,  Ergonomie | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| Import/Export | F - Richtigkeit | Manueller Test, autom. test | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| Bestellung | F - Richtigkeit | Manueller Test, autom. test | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| Email Versand | F - Richtigkeit | Manueller Test, autom. test | nach jeder Änderung,  Meilenstein,  am Ende | Teammitglied, Anwender |  |
| **nicht funktionale Eigenschaften / Anforderungen** | | | | | |
| generell | Zuverlässigkeit | Lasttest | am Ende | Teammitglied |  |
| generell | Benutzbarkeit | Kompatibilitätstest | Meilenstein | Teammitglied |  |
| generell | Effizienz | Performanztest | Am Ende | Teammitglied |  |
| optional | ….. |  |  |  |  |

# Testdurchführung – ein Prozessüberblick

Der vorliegende Abschnitt beschreibt das Template, nach dem die einzelnen Testfälle beschrieben werden sollen. Die Testfälle basieren auf dem zuvor verabschiedeten Konzept (Pflichtenheft, tech. Spezifikation). Sie bilden die Grundlage für den späteren Test und anschließende Abnahme des Systems.

Durch den Test wird sichergestellt, dass mit Hilfe der erstellten Software die spezifizierten Geschäftsprozesse fachlich korrekt und den Anforderungen entsprechend durchgeführt werden können. Testen bedeutet somit den Abgleich zwischen den dokumentierten Anforderungen und den vorliegenden Modulen durchzuführen.

Ein Test ist dann erfolgreich, wenn er dazu beiträgt, Fehler zu beseitigen bzw. nicht erfüllte Anforderungen zu erkennen.

Der abgearbeitete Testplan soll einerseits als Grundlage für das Abnahme-Testat des Auftraggebers dienen und andererseits sicherstellen, dass nur ein getestetes und fehlerfreies System in Produktion genommen wird.



Abbildung 3: Testorganisation und –ablauf

Eine Spezifikation eines Testfalls erfolgt in 2 Schritten:

1. Definition des logischen Testfalls,
2. Erweiterung durch tatsächliche Eingabewerte zu einem konkreten Testfall.

Ein Testfall ist eine Kombination von Eingabedaten, Randbedingungen und erwarteten Ausgaben, die einem bestimmten Zweck dienen. Man prüft z.B., ob Vorgaben in einem Spezifikationsdokument eingehalten werden oder ob der Programmablauf tatsächlich dem erwarteten Pfad entspricht.   
  
Bestandteile eines Testfalls sollten sein:

* Testfall ID   
  Jeder Testfall ist mit einer eindeutigen Referenz zu versehen.
* Zu testende Funktionen   
  Hier sind alle Testobjekte / Funktionen zu beschreiben, die von diesem Testfall ausgeführt werden. Testobjekte können z.B. separate Softwaremodule oder einzelne Webseiten sein.
* Eingaben   
  Hier sind alle Eingaben zu spezifizieren, die der Tester braucht, um den Testfall auszuführen.   
    
  Die Eingaben können sowohl als Wert angegeben werden (ggf. mit Toleranzen) als auch als Name, falls es sich um konstante Tabellen oder um Dateien handelt. Außerdem sind alle betroffenen Datenbanken, Dateien, Terminalmeldungen, nicht flüchtige Speicherbereiche und vom Betriebssystem übergebene Werte anzugeben.
* Erwartetes Verhalten  
  Hier sind alle Ausgaben oder Verhalten anzugeben, die aufgrund der Tests zu erwarten sind.   
  Für jede erwartete Ausgabe/Verhalten muss der genaue erwartete Wert/Ergebnis (ggf. mit Toleranzen) angegeben werden. Falls erforderlich, sind auch weitere erwartete Eigenschaften der Ausgabe (z.B. Antwortzeit) anzugeben.
* Testvoraussetzung  
  Manche Testfälle erfordern spezielle Ausführungsvorschriften, die nur für diesen einen Testfall gelten. In diesem Fall ist es nicht zweckmäßig, die Ausführungsvorschrift im Testskript/Programm unterzubringen. Stattdessen wird sie an dieser Stelle definiert.

# Testfälle

## Testfallbeschreibung

Pro Testfall soll das folgende Template angewandt werden:

| **Testfall** | **Beschreibung** |
| --- | --- |
| Testfall-Nummer | * Alle Testfälle sollen aufsteigend durchnummeriert werden |
| Testart | * zu unterscheiden sind die Testarten: * Funktionstest * Integrationstest * Performancetest * Ergonomie |
| Zu testender Geschäftsprozess/  Zu testende Funktionsgruppe | * z.B. Import einer CSV Datei, Email-Versand, etc. |
| Testziel | * Kurze verbale Ziel-Beschreibung des Testfalls |
| Testvoraussetzungen | * Beschreibung von Systemvoraussetzungen, die erfüllt sein müssen, bevor der Test durchgeführt wird. Insbesondere müssen hier die Daten beschrieben werden, die im System oder in einer vorhandenen Datei bereits hinterlegt sein müssen |
| Testfalldaten | * Detaillierte Beschreibung des Testfalls auf Datenebene: * Inhaltliche Testfalldaten * ggf. wie wird die Funktion aufgerufen/ausgelöst |
| Erwartetes Verhalten | * Welche Ergebnisse muss das System bei dem vorliegenden Testfall liefern (Was soll rauskommen?). |

**Testergebnis**

Folgendes Template soll das Testergebnis jedes einzelnen Testfalls dokumentieren:

|  |  |  |
| --- | --- | --- |
| Testergebnis | □ Bestanden □ Nicht Bestanden | |
| Fehlerkategorie | □ Leicht □ Mittel □ Schwer[[1]](#footnote-1) | |
| Bemerkung | z.B. beobachtetes Verhalten, Fehlerbeschreibung, Anregung oder Verbesserungsvorschlag etc. | |
| Tester Kunde | Tester Auftragnehmer | Datum |
|  |  |  |

## Beispiel eines Testfalls: Import einer CSV-Datei In das GUI Tool

| **Testfall** | **Beschreibung** |
| --- | --- |
| Testfall-Nummer | * 00001 |
| Testart | * Funktionstest |
| Zu testender Geschäftsprozess/  Zu testende Funktionsgruppe | * Import eines Fremdformates |
| Testziel | * Validieren, ob eine CSV-Datei aus einem Fremdsystem richtig und vollständig inkl. aller Feldtypen importiert und für die weitere Verarbeitung aufbereitet wird |
| Testvoraussetzungen | * Gültige Installation * Access oder Access-Importfilter muss installiert sein |
| Testfalldaten | * Siehe Datei customer.dbf  | ***Feldname*** | ***Feldinhalt*** | ***Typ*** | | --- | --- | --- | | Aenderung\_LfdNr | 000000029 | N(10) | | Haushaltsnr | 000000021 | N(10) | | Aenderungs\_KZ | 0 | C(1) | | Zugangsdatum | 01.01.2000 | D(8) | | GueltigkeitAbDatum | 10.01.2000 | D(8) | | Anrede\_KZ | 1 | N(1) | | Titel\_KZ | 0 | N(1) | | Vorname | Kurt | C(30) | | Nachname | Kruel | C(30) | | Nachname\_Neu |  | C(30) | | Strasse\_Alt | Ahornallee 250 | C(40) | | PLZ\_Alt | 33330 | C(5) | | Ort\_Alt | Gütersloh | C(30) | | Strasse\_Neu | Ahornallee 252 | C(30) | | PLZ\_Neu | 33330 | C(5) | | Ort\_Neu | Gütersloh | C(30) | | TelefonNr |  | C(18) | | Mobilfunk\_KZ |  | N(1) | | Quellen\_KZ | 1 | N(2) | | Bearbeitungsdatum |  | D(8) | | Bearbeitungsstatus | 1 | C(1) | | Wiedervorlagezaehler | 0 | N(2) | | Bearbeitungsinfo |  | C(128) | |
| Erwartetes Verhalten | * Datei ist vollständig und korrekt übernommen, alle Felder sind innerhalb des GUIs korrekt separiert und erkennbar, der Delimitertyp ist Semikolon und der Zeichensatz ist korrekt von DOS zu Unicode konvertiert. |

|  |  |  |
| --- | --- | --- |
| Testergebnis | □ Bestanden X Nicht Bestanden | |
| Fehlerkategorie | □ Leicht X Mittel □ Schwerwiegend | |
| Bemerkung | Format wird zwar importiert, aber die Umlaute nicht korrekt umgesetzt | |
| Tester Kunde | Tester Auftragnehmer | Datum |

## Beispiel eines Testfalls: M/W Erkennung durch den Namen

| **Testfall** | **Beschreibung** |
| --- | --- |
| Testfall-Nummer | * 00002 |
| Testart | * Funktionstest |
| Zu testender Geschäftsprozess/  Zu testende Funktionsgruppe | * Komponenten (analyse, refctrl, formatter, virtualmachine, kernel ) Interaktion funktionieren ohne Konflikte. |
| Testziel | * Validieren, ob eine CSV-Datei aus Access richtig und vollständig inkl. aller Feldtypen importiert und in CSV für die weitere Verarbeitung aufbereitet wird |
| Testvoraussetzungen | * Gültige Installation * Templates + minimal reference data + TCF eingerichtet |
| Testfalldaten | „Mary Schmidt“ |
| Erwartetes Verhalten | Erkennung weiblich. |

|  |  |  |
| --- | --- | --- |
| Testergebnis | X Bestanden □ Nicht Bestanden | |
| Fehlerkategorie | □ Leicht □ Mittel □ Schwerwiegend | |
| Bemerkung | Format wird zwar importiert, aber die Umlaute nicht korrekt umgesetzt | |
| Tester Kunde | Tester Auftragnehmer | Datum |

## Beispiel eines Testfalls: GUI-Kommunikation

| **Testfall** | **Beschreibung** |
| --- | --- |
| Testfall-Nummer | * 00003 |
| Testart | * Integrationstest |
| Zu testender Geschäftsprozess/  Zu testende Funktionsgruppe | * Soap Commandos |
| Testziel | * Validieren, ob Soap Server und GUI korrekt kommunizieren |
| Testvoraussetzungen | * Gültige Installation von GUI Framework und Soap Server |
| Testfalldaten | * Beliebige Projektdatei, die vom GUI erzeugt wurde. |
| Erwartetes Verhalten | * Aufruf von Run startet die Verarbeitung * Aufruf von Cancel stoppt die Verarbeitung * Aufruf von Status liefert eine Satusmessage. |

|  |  |  |
| --- | --- | --- |
| Testergebnis | □ Bestanden X Nicht Bestanden | |
| Fehlerkategorie | □ Leicht □ Mittel X Schwerwiegend | |
| Bemerkung | Nach Cancel stürzt das GUI mit einer Unhandled Exception ab | |
| Tester Kunde | Tester Auftragnehmer | Datum |

# Testprotokoll

| **TestfallNr.** | **Datum** | **Status** | **Schweregrad** | **Datum 2. Lauf** | **Status 2. Lauf** |
| --- | --- | --- | --- | --- | --- |
| 01 | 15.01.13 | bestanden |  |  |  |
| 02 | 15.01.13 | bestanden |  |  |  |
| 03 | 15.01.13 | bestanden |  |  |  |
| 04 | 15.01.13 | nicht bestanden | mittel | 20.01.13 | bestanden |
| 05 | 15.01.13 | bestanden |  |  |  |
| 06 | 15.01.13 | nicht bestanden | schwer | 20.01.13 | bestanden |
| 07 | 16.01.13 | bestanden |  |  |  |
| 08 | 16.01.13 | bestanden |  |  |  |
| 09 | 16.01.13 | bestanden |  |  |  |
| 10 | 16.01.13 | nicht bestanden | leicht | 20.01.13 | bestanden |
| 11 | 16.01.13 | bestanden |  |  |  |

# Anhang

## Fehlerkategorien

Für die Abnahme des Systems sind folgende Fehlerklassen definiert:

* **3 = Schwerer Mangel** Produktivsetzung nicht möglich (Nachhaltige Störung des  
   Softwareablaufes mit daraus resultierender Funktionsuntüch-  
   tigkeit des Systems bzw. Störung von Systemteilen, die zur  
   Störung aller Arbeitsabläufe beim Auftraggeber führt.)
* **2 = Mittlerer Mangel** Produktivsetzung möglich aber mangelhafte Funktionen nicht   
   nutzbar (Durch eine Störung treten in Teilen der Programm-  
   abläufe nicht unerhebliche Störungen auf, so dass Teile der   
   Software nicht verwendbar sind.)
* **1 = Leichter Mangel** Produktivsetzung durch Workaround mit vertretbarem Zu-  
   satzaufwand möglich (Alle anderen als die in den vorstehen-  
   den Prioritätsgraden beschriebenen Störungsbilder)

## Q-Kriterien ISO 9126

| **Gruppe** | **Q-Kriterium** | |
| --- | --- | --- |
| **Funktionalität** | | |
| Sind alle im Pflichtenheft aufgeführten Kriterien vorhanden und ausführbar? | **Angemessenheit** | Merkmale von Software, die sich auf das Vorhandensein und die Eignung einer Menge von Funktionen für spezifizierte Aufgaben beziehen. |
| **Richtigkeit** | Merkmale von Software, die sich beziehen auf das Liefern der richtigen oder vereinbarten Ergebnisse oder Wirkungen. |
| **Interoperabilität** | Merkmale von Software, die sich auf ihre Eignung beziehen, mit vorgegebenen Systeme zusammenzuwirken. |
| **Ordnungsmäßigkeit** | Merkmale von Software, die bewirken, dass die Software anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen oder ähnliche Vorschriften erfüllt. |
| **Sicherheit** | Merkmale von Software, die sich auf ihre Eignung beziehen, unberechtigten Zugriff, sowohl versehentlich als auch vorsätzlich, auf Programme und Daten zu verhindern. |
| **Zuverlässigkeit** | | |
| Zu welchem Grad erfüllt die Software dauerhaft und korrekt die geforderten Funktionen? | **Reife** | Merkmale von Software, die sich auf die Häufigkeit von Versagen durch Fehlzustände in der Software beziehen. |
| **Fehlertoleranz** | Merkmale von Software, die sich auf ihre Eignung beziehen, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren. |
| **Wiederherstellbarkeit** | Merkmale von Software, die sich beziehen auf die Möglichkeit, bei einem Versagen ihr Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen, und auf die dafür benötigte Zeit und den benötigten Aufwand. |
| **Benutzbarkeit** | | |
| Wie schnell kann man den Umgang mit der Software lernen und wie leicht ist sie zu bedienen? | **Verständlichkeit** | Merkmale von Software, die sich auf den Aufwand für den Benutzer beziehen , das Konzept und die Anwendung zu verstehen. |
| **Erlernbarkeit** | Merkmale von Software, die sich auf den Aufwand für den Benutzer beziehen, ihre Anwendung zu erlernen. (z.B. Ablaufsteuerung, Eingabe, Ausgabe) |
| **Bedienbarkeit** | Merkmale von Software, die sich auf den Aufwand für den Benutzer bei der Bedienung und Ablaufsteuerung beziehen. |
| **Effizienz** | | |
| Wie sind zeitliches Verhalten und Ressourcenverbrauch bei gegebenen System-voraussetzungen? | **Zeitverhalten** | Merkmale von Software, die sich beziehen auf die Antwort- und Verarbeitungszeiten und auf den Durchsatz bei der Ausführung ihrer Funktionen. |
| **Verbrauchsverhalten** | Merkmale von Software, die sich darauf beziehen, wie viele Betriebsmittel bei der Erfüllung ihrer Funktionen benötigt werden und wie lange. |
| **Änderbarkeit** | | |
| Mit welchem Zeit- und Arbeitsaufwand lassen sich Änderungen sowie Fehlererkennung und -behebung durchführen? | **Analysierbarkeit** | Merkmale von Software, die sich auf den Aufwand beziehen, der notwendig ist, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen. |
| **Modifizierbarkeit** | Merkmale von Software, die sich auf den Aufwand beziehen, der zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder zur Anpassung an Umgebungsänderungen notwendig ist. |
| **Stabilität** | Merkmale von Software, die sich auf das Risiko unerwarteter Wirkungen von Änderungen beziehen. |
| **Prüfbarkeit** | Merkmale von Software, die sich auf den Aufwand beziehen, der zur Prüfung der geänderten Software notwendig ist. |
| **Übertragbarkeit** | | |
| Mit welchem Aufwand lässt sich die Software an geänderte/ verbesserte Systembedingungen anpassen bzw. in neuen Systemen einsetzen? | **Anpassbarkeit** | Merkmale von Software, die sich auf die Möglichkeit beziehen, sie an verschiedene festgelegte Umgebungen anzupassen, wenn nur Schritte unternommen oder Mittel eingesetzt werden, die für diesen Zweck für die betrachtete Software vorgesehen sind. |
| **Installierbarkeit** | Merkmale von Software, die sich auf den Aufwand beziehen, der zur Installation der Software in einer festgelegten Umgebung notwendig ist. |
| **Konformität** | Merkmale von Software, die bewirken, dass die Software Normen oder Vereinbarungen zur Übertragbarkeit erfüllt. |
| **Austauschbarkeit** | Merkmale von Software, die sich beziehen auf die Möglichkeit, diese anstelle einer anderen Software in der Umgebung jener Software zu verwenden und auf den dafür notwendigen Aufwand. |

## Q-Kriterien für Dokumente

Für die Erreichung des Projektzieles, das Produkt „Dokument“ zu erzeugen, dass den fachlichen und technischen Anforderungen des Auftraggebers entspricht, ergeben sich z.B. die folgenden Qualitätsmerkmale:

| **Merkmal** | **Erläuterung** | **Mindest-anfordrg.** | **Prüfmöglichkeit** |
| --- | --- | --- | --- |
| **Eindeutigkeit** | Eignung von Dokumenten zur unmissverständlichen Vermittlung von Informationen für jeden Leser |  | Keine offenen Fragen zu den einzelnen Abschnitten (Prüfung durch Gruppeninspektion und Diskussion) |
| **Lesbarkeit** | Eignung von Dokumenten zur Entnahme der darin enthaltenen Informationen | ja | Prüfung durch Einsatz eines unbedarften Testlesers, Vorhandensein eines Glossars, Erläuterung von Fachbegriffen |
| **Verständlichkeit** | Eignung von Dokumenten zur erfolgreichen Vermittlung der darin enthaltenen Informationen an einen sachkundigen Leser | ja | Vorhandensein eines Glossars, Integration von Illustrationen, Diagrammen |
| **Detaillierungsgrad** | Vorhandensein der ausreichenden Beschreibung der fachlichen und technischen Einzelheiten im Dokument |  | Beschreibung der Sonder- und Ausnahmefälle, gleiche Behandlung (gleiche Detaillierung) aller Textabschnitte |
| **Funktionale Vollständigkeit** | Vorhandensein der für den Zweck der Dokumentation notwendigen und hinreichenden Information | ja | Einsatz des <KUNDE>Templates gewährleistet die Voll-ständigkeit an notwendigen Informationen, Beschreibung der Sonder- und Ausnahmefälle |
| **Fehlerfreiheit** | Nichtvorhandensein von sprachlichen Fehlern, die die Informationsaufnahme beeinträchtigen |  | Rechtschreib- und Grammatikprüfung |
| **Widerspruchsfreiheit** | Nichtvorhandensein von einander entgegenstehenden Aussagen im Dokument |  | Unnötige Redundanzen sollen vermieden werden, Dokument soll in sich konsistent sein |
| **Aktualität** | Übereinstimmung der Beschreibung der Situation in Dokument und Wirklichkeit |  | Gespräche mit dem Auftraggeber (Kundeninspektion, Workshops) |
| **Funktionale Korrektheit** | Nichtvorhandensein von funktionalen Fehlern, die den fachlichen und technischen Inhalt betreffen | ja | Wiedergabe der Anforderungen aus dem Vorgängerdokument |
| **Normenkonformität** | Erfüllung der für die Erstellung von Dokumenten geltenden Vorschriften und Normen |  | Einsatz des <KUNDE>Templates gewährleistet die formale Richtigkeit |
| **Änderbarkeit** | Eignung von Dokumenten zur Ermittlung aller von einer Änderung betroffenen Dokumententeile und zur Durchführung der Änderung |  | Einsatz des <KUNDE>Templates gewährleistet die formale Änderbarkeit, unnötige Redundanzen sollen vermieden werden |

1. Die Beschreibung der Fehlerkategorien entnehmen Sie bitte dem beigefügten Anhang [↑](#footnote-ref-1)