**CHECKLISTE**

**TESTDESIGNTECHNIKEN**

Inhaltsverzeichnis

Inhaltsverzeichnis 1

Checkliste Testdesigntechniken 2

Datenkombinationstest (DKT) 2

Datenzyklustest (DZT) 2

Entscheidungstabellentest (ETT) 2

Elementarer Vergleichstest (EVT) 3

Error Guessing (EG) 3

Exploratives Testen (ET) 3

Geschäftsprozesstest (GPT) 3

Real-Life-Test (RLT) 4

Semantischer Test (SEM) 4

Syntaktischer Test (SYN) 5

Anwendungsfalltest (AFT) 6

# 

## Datenkombinationstest (DKT)

Der Datenkombinationstest (DKT) ist eine flexible Technik zum Testen der Funktionalität sowohl auf der Detail- als auch auf der Gesamtsystemebene. Für den DKT ist keine spezifische Testbasis erforderlich. Alle Informationen über die Funktionalität können herangezogen werden:

* Formale Systemdokumentation, z.B. funktionales Design, logisches Datenmodell und Anforderungen
* Informelle Dokumentation, z.B. Handbücher, Prospekte, Voruntersuchungen und Memos
* Erfahrungen aus dem Fachbereich, die nicht dokumentiert, sondern “in den Köpfen der Experten” vorhanden sind

## Datenzyklustest (DZT)

Der Datenzyklustest hat nicht vorrangig zum Ziel, funktionale Fehler in einzelnen Funktionen zu verfolgen, sondern Integrationsfehler zu finden. Der Schwerpunkt des Tests liegt auf der Verbindung zwischen verschiedenen Funktionen und der Art, wie sie mit gemeinsam genutzten Daten umgehen. Der DZT ist am effektivsten, wenn die Funktionalität der einzelnen Funktionen bereits ausreichend getestet wurde. Die wichtigste Testbasis ist hier die CRUD-Matrix und eine Beschreibung der anwendbaren Integritätsregeln. Sollte keine Matrix verfügbar sein, sollte sie im Rahmen des Entwurfs des Testdesigns für den Datenzyklustest erstellt werden. Die folgende Checkliste kann dazu genutzt werden, um die vorliegende Testbasis auf die Verwendung für den DZT hin zu prüfen:

* Existiert eine CRUD-Matrix?
* Kann jeder Datensatz angelegt, aufgerufen, geändert und gelöscht werden?
* Ist eindeutig, durch welche Funktion(en) der Datensatz angelegt, aufgerufen, geändert und gelöscht werden kann?
* Sind alle Datensätze beschrieben?
* Existiert ein Datensatzdiagramm (Daten-Beziehungs-Diagramm, DBD)?
* Sind die Beziehungen zwischen den verschiedenen Datensätzen beschrieben?
* Wurden Prüfungen für die Verbindungen beschrieben?

## Entscheidungstabellentest (ETT)

Der Entscheidungstabellentest ist eine intensive Technik zum Testen einer Detailfunktionalität. Die erforderliche Testbasis enthält Bedingungen oder Entscheidungstabellen. Die Art und Struktur dieser Testbasis ist für die Anwendung der Entscheidungstabellentechnik von geringer Bedeutung. Ziel des Entscheidungstabellentests ist eine intensive Abdeckung der Bedingungen und nicht der funktionalen Pfade. Die folgenden Punkte können für die Spezifikation der Designtechnik genutzt werden:

* Sind die prozessbeeinflussenden Bedingungen leicht erkennbar und wurden sie beschrieben?
* Ist das Ergebnis des Prozesses klar und ist es möglich, Voraussagen zu treffen?

## Elementarer Vergleichstest (EVT)

Der elementare Vergleichstest ist eine intensive Technik zum umfassenden Testen der Funktionalität. Die erforderliche Testbasis ist der Pseudocode oder eine vergleichbare Spezifikation, in der die Entscheidungspunkte und funktionalen Pfade ausführlich und strukturell definiert sind. Der EVT zielt auf eine intensive Abdeckung der Entscheidungspunkte und nicht auf die Kombination funktionaler Pfade ab. Folgende Checkliste sollte bei der Prüfung der Spezifikation in Betracht gezogen werden:

* Wurde die angewandte Methode so beschrieben, das verschiedene funktionale Pfade klar erkennbar sind?
* Ist klar, in welchem Zustand jeder funktionale Pfad genommen werden kann?
* Wurde die Verarbeitung eindeutig beschrieben, inklusive der Ein- und Ausgabewerte?
* Wurde die Verarbeitung für jedes Objekt beschrieben?

## Error Guessing (EG)

Ein spezifisches Review eines bestimmten Aspektes der Testbasis ist kein notwendiger Bestandteil zum Gebrauch der Testtechnik Error Guessing. Genau wie exploratives Testen ist Error Guessing eine eher untypische Technik unter den Testdesigntechniken. Es basiert auf keiner der beschriebenen Grundtechniken. Es ist ein gründliches Verständnis des zu testenden (Sub) Systems erforderlich, welches aus der Dokumentation ersichtlich werden sollte. Diese detaillierte Einsicht kann zusätzlich aus der Durchführung von strukturierten Tests in Kombination mit Error Guessing gewonnen werden.

## Exploratives Testen (ET)

Wie Error Guessing lässt sich exploratives Testen nur schwerlich mit den übrigen Testdesigntechniken vergleichen. Es basiert auf keiner der beschriebenen Grundtechniken, lässt dem Tester frei Wahl in der anzuwendenden Grundtechnik und bietet keine garantierte Abdeckung. Exploratives Testen wird oft mit dem Testen ohne formale Testbasis, wie z.B. einem funktionalen Design, in Verbindung gebracht. Das trifft aber nicht unbedingt zu. Es ist absolut möglich, die Technik auf ein gut beschriebenes System anzuwenden. Generell eignet sich die Technik allerdings sehr gut für Situationen, in denen keine dokumentierte Testbasis vorhanden ist. Exploratives Testen legt den Schwerpunkt weniger auf eine dokumentierte Testbasis als vielmehr auf andere Möglichkeiten, die Zweckmäßigkeit oder Eignung des Testobjekts festzustellen, z.B. indem man sich im Verlauf der Testdurchführung mit dem System vertraut macht.

## Geschäftsprozesstest (GPT)

Der Geschäftsprozesstest ist eine Technik, die insbesondere auf das Testen des Qualitätsmerkmals “Angemessenheit” (Integration zwischen der Betriebsführung und dem automatisierten Informationssystem) angewandt wird. Die Testbasis sollte strukturierte Informationen über das erforderliche Systemverhalten in Form von Pfaden und Entscheidungspunkten beinhalten. Die Testbasis für den Geschäftsprozesstest besteht hauptsächlich aus Prozessbeschreibungen und ähnlichen Formularen. Um den Geschäftsprozesstest anzuwenden, ist ein Prozessdiagramm notwendig. Die folgende Checkliste kann verwendet werden, um bestehende Prozessabläufe zu prüfen:

* Sind alle manuellen Abläufe, die der User ausführt, beschrieben worden?
* Wurde die Verantwortlichkeit und Befugnis für jeden manuellen Ablauf beschrieben?
* Wurde eine Beschreibung für individuelle Abläufe erstellt?
* Sind die sicherheitsrelevanten Aspekte der Abläufe beschrieben worden?
* Wurde das “auslösende Ereignis” klar dargestellt? (Wann sollte der Ablauf gestartet werden?)
* Wurde aufgezeigt, welche Art von Daten (Formulare) benutzt werden sollen und wo diese zu erhalten sind?
* Wurden die durchzuführenden Aktivitäten, inklusive der Ausnahmen und Prüfungen, beschrieben?
* Ist das Ergebnis jedes Vorgangs klar definiert?
* Wurden die unterschiedlichen Entscheidungspunkte beschrieben, inklusive der zugehörigen Zustände?
* Wurde zwischen Endnutzer- und Systemnutzerabläufen unterschieden?
* Wurden die Zusammenhänge zwischen den automatisierten und nicht-automatisierten Systemteilen ausführlich beschrieben?
* Ist das Benutzerhandbuch (der Entwurf) verfügbar?

## Real-Life-Test (RLT)

Mit dem Real-Life-Test (RLT) soll nicht das Systemverhalten in getrennten Situationen getestet werden, sondern es soll der realistische Gebrauch des Systems statistisch aussagekräftig simuliert werden. Dieser Test konzentriert sich hauptsächlich auf Merkmale wie z.B. Effektivität, Verbindungsfähigkeit, Kontinuität und Leistungsfähigkeit des zu testenden Systems. Um testen zu können, ob ein System einer realistischen Nutzung standhalten kann, muss diese Nutzung zunächst spezifiziert werden. Dies dient dann auch als Testbasis und wird in diesem Zusammenhang oft als Profil bezeichnet. Die beiden häufigsten Arten sind das Anwendungs- und Lastprofil. Die folgenden Fragen stellen einen wichtigen Teil des Grundgerüstes eines Reviews des Real-Life-Tests dar:

* Wurde die erwartete Häufigkeit der Nutzung auf einem funktionalen Level beschrieben?
* Wurde für jeden Nutzertyp beschrieben, welche Funktionen von ihm ausgeführt werden?
* Wurde für jede Funktion beschrieben, wann sie ausgeführt werden soll (täglich, wöchentlich, jährlich, tagsüber, abends)?
* Gibt es Einblick in die Beziehungen zwischen den verschiedenen systemübergreifenden Batch-Prozeduren?
* Wurde die Konfiguration für die Produktionsumgebung dokumentiert (Hardware, Netzwerk, Systemsoftware, DBMS, etc.)?
* Wurden spezifische Anforderungen an die Onlinefunktionalität gestellt?
* Gibt es einen zeitlichen Unterschied zwischen der Ausführung einer Funktion und der Bildschirmausgabe?
* Wurden spezielle Anforderungen an die Schnelligkeit von Datenwiederherstellung und Manipulation erhoben?
* Wurden spezifische Anforderungen an die Performance der Batch-Funktionalität festgelegt?
* Wurden spezielle Anforderungen an den Speicherverbrauch festgelegt?
* Gibt es Anforderungen über die Anzahl der Datenbankabfragen pro Vorgang?
* Gibt es Anforderungen in Bezug auf die maximale Größe der Seiten und des Buffers?
* Existieren Anforderungen bezüglich der Größe der Anwendung und/oder der Datenbank?

## Semantischer Test (SEM)

Die Testbasis besteht aus den semantischen Regeln, die spezifizieren, womit ein Datenelement konform sein sollte, um vom System als gültige Eingabe akzeptiert zu werden. Semantische Regeln sind mit *Beziehungen zwischen Daten* verbunden. Diese Beziehungen können zwischen den Daten eines Bildschirms, zwischen denen in verschiedenen Bildschirmen und zwischen Eingabe- und vorhandenen Daten in einer Datenbank bestehen. Semantische Regeln lassen sich in verschiedenen Dokumenten aufstellen, üblich sind aber die Folgenden: Funktionale Spezifikationen der relevanten Funktion oder des Eingabebildschirms und Geschäftsregeln, die auf die Funktionen insgesamt anwendbar sind. Die folgende Liste kann zur Prüfung der Spezifikationen genutzt werden:

* Wurden Standards für die Fehlerbehandlung auf (Sub) System-Level festgelegt?
* Wurden die Eingabeprüfungen (insbesondere die Verbindungsprüfungen) , inklusive der zugehörigen Fehlermeldung, als Teil der funktionalen Beschreibung dokumentiert und können diese implementiert werden?
* Werden besondere Anforderungen an die Zugriffssicherheit auf die Funktionen und/oder Daten gestellt?
* Wurden die Benutzerprofile in Bezug auf die dokumentiert?
* Ist bekannt, welche Anforderungen an die Identifikation (User-ID) und die Authentifizierung (Passwort) gestellt werden?

## Syntaktischer Test (SYN)

Die Testbasis für den syntaktischen Test besteht aus den syntaktischen Regeln, die spezifizieren, was ein Attribut erfüllen sollte, um vom System als gültige Ein-/Ausgabe akzeptiert zu werden. Diese Regeln beschreiben den Wertebereich für das relevante Attribut. Wird ein außerhalb dieses Bereichs liegender Wert als Attribut angeboten, sollte das System die Verarbeitung auf kontrollierte Weise – normalerweise mit einer Fehlermeldung - abbrechen. Syntaktische Regeln lassen sich in verschiedenen Dokumenten aufstellen, normalerweise werden sie aber in Folgenden beschrieben: Im “Datenwörterbuch” und anderen Datenmodellen, in denen die Eigenschaften aller Daten beschrieben sind und den funktionalen Spezifikationen der relevanten Funktion bzw. des Eingabebildschirms, in denen die Anforderungen an die Attribute spezifiziert sind. Die folgende Checkliste kann dafür genutzt werden:

* Sind geeignete Standards auf Systemebene beschrieben worden?
* Sind geeignete Standards auf Teilsystemebene beschrieben worden?
* Wurden die Layouts der Bildschirme, Gestaltung und der Dialoge beschrieben?  
  Wurden in diesem Zusammenhang die folgenden Punkte beachtet:
  + Feldlänge der Elemente;
  + Position der Elemente auf dem Bildschirm;
  + Dem Unterschied zwischen Ein-/Ausgabeelementen;
  + Primäre Eingabeprüfungen (die nicht im Wertebereich beschrieben sind);
  + Fehlerbehandlung;
  + Verbindliches/unverbindliches Attribut
  + Mögliche Funktionstasten, Hilfebildschirme und Auswahlmöglichkeiten?
* Wurden die “Bildschirmelemente” und/oder die Attribute im Datenmodell eingefügt?
* Sind die Arten (numerisch, alphanumerisch, Datum) und die Wertebereiche der Ein-/Ausgaben beschrieben worden?
* Sind die spezifizierten und nicht-spezifizierten Datenelemente konsistent mit den Optionen aus dem Datenmodell?
* Halten die beschriebenen Bildschirmlayouts die Standards ein?
* Wurden die Layouts der Berichte beschrieben?  
  Ist in diesem Zusammenhang an die folgenden Punkte gedacht worden:
  + Feldlänge der Eingaben;
  + Position der Eingaben im Bericht?
* Wurden die “Berichtseingaben” und/oder die Attribute im Datenmodell aufgenommen?
* Entsprechen die beschriebenen Berichtslayouts den Standards?

## Anwendungsfalltest (AFT)

Der detaillierte Inhalt einer Checkliste zur Ermittlung, ob ein Anwendungsfall für einen Anwendungsfalltest geeignet ist, hängt davon ab, wie der Anwendungsfall beschrieben ist. Nachfolgend einige Prüfpunkte, die als Grundlage zum Erstellen einer Checkliste herangezogen werden können:

* Ist die Anwendungsfallvorlage (Standard für das Projekt/Unternehmen) vollständig ausgefüllt?
* Ist ein Anwendungsfalldiagramm verfügbar?
* Ist der Anwendungsfall in sich abgeschlossen?
* Ist das Ziel des Anwendungsfalls klar?
* Ist klar, für welche Akteure der Anwendungsfall gedacht ist?
* Bezieht sich der Anwendungsfall auf die Funktionalität (und nicht auf die Bildschirmabfolge)?
* Wurden alle vorhersehbaren Alternativen beschrieben?
* Wurden alle bekannten Ausnahmen beschrieben?
* Enthält der Anwendungsfall einen vollständigen schrittweisen Plan?
* Wurde jeder Schritt in den Szenarien klar, eindeutig und vollständig beschrieben?
* Werden im Anwendungsfall alle für die Durchführung der Aufgabe relevanten Akteure und Schritte aufgeführt?
* Sind die beschriebenen Schritte ausführbar?
* Ist das Ergebnis der Schritte nachprüfbar?
* Entsprechen die Vor- und Nachbedingungen dem Anwendungsfall?