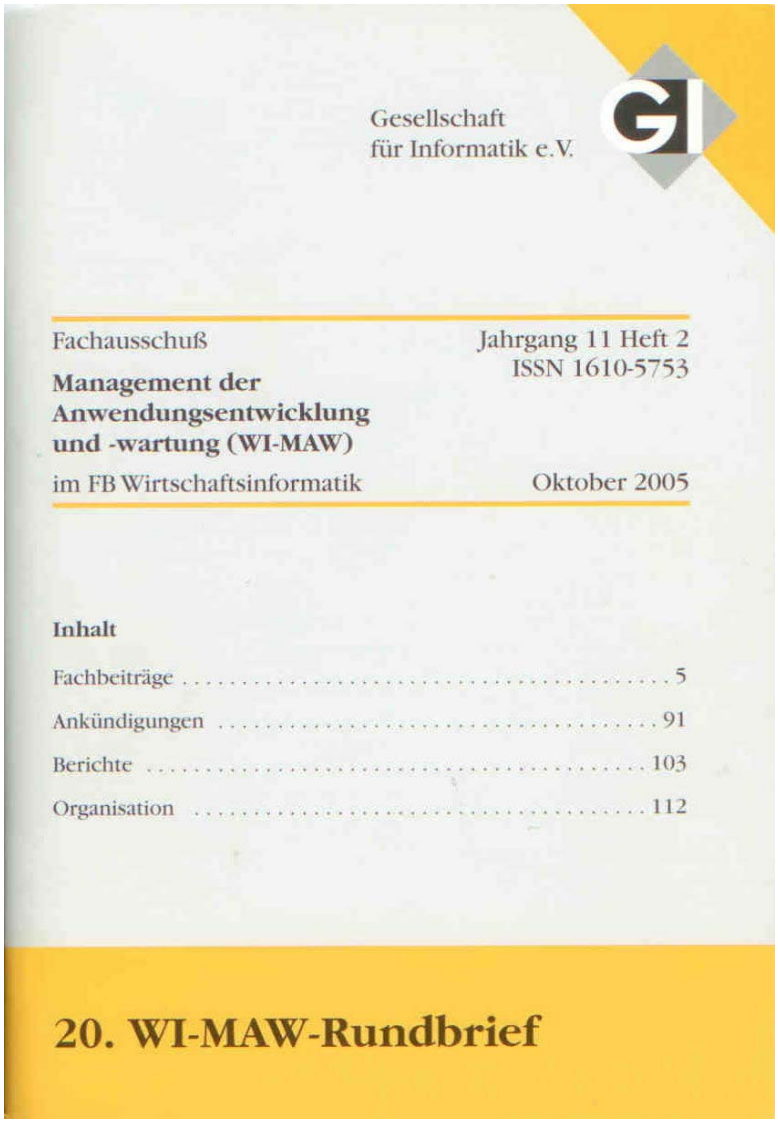
Durchführung eines Anwendertests in der Praxis
- Pragmatische Ansätze und Tools zur Testplanung, Teststeuerung und Fehlerkommunikation -

Dipl. Wi.-Ing. Daniel Sladic
IFS Consulting GmbH & Co. KG
Daniel.Sladic@IFS-Consulting.com
http://www.ifs-consulting.com

erschienen in:



Gesellschaft
für Informatik e.V.

Fachausschuß
Management der Anwendungsentwicklung und -wartung (WI-MAW)
im FB Wirtschaftsinformatik

Jahrgang 11 Heft 2
ISSN 1610-5753
Oktober 2005

Inhalt

Fachbeiträge 5
Ankündigungen 91
Berichte 103
Organisation 112

20. WI-MAW-Rundbrief

Inhaltsverzeichnis

1 Methodische Betrachtungen zum Testansatz und Begriffsbestimmungen 4
2 Unterscheidung der Testphasen 4
3 Vorbereitende Schritte 6
3.1 Testvorbereitung und Testorganisation 6
3.1.1 Definition von Testfällen und Testskripten 7
3.1.2 Erstellung und Verwendung von Testskripten 7
3.1.3 Verwaltung der Testskripte und Vorbereitung der Testplanung 9
4 Testplanung, Ausführung und Statuscontrolling 10
4.1 Testplanung 10
4.2 Testausführung 11
4.2.1 Fehlermanagement 11
4.2.1.1 Fehlerkategorien 12
4.2.1.2 Fehlererfassung und -kommunikation 13
4.3 Statustracking und -reporting 16
5 Testmanagement 17
5.1 Aufgaben der Testleitung in der Vorbereitungsphase 17
5.2 Aufgaben der Testleitung während der Testdurchführung 18
6 Testdokumentation 18
6.1 Bestandteile der Testdokumentation 18
6.1.1 Das Testkonzept 19
6.1.2 Testspezifikation 20
6.1.3 Testprotokolle und Abschlussbericht 20
7 Zusammenfassung / Fazit 21

Abbildungsverzeichnis

Abbildung 1: Schematische Darstellung der Testphasen 5
Abbildung 2: Testmodell und Testorganisation im Anwendertest 6
Abbildung 3: Übersicht der definierten Testskripte 7
Abbildung 4: Testskript – Beispiel für Aufbau und Handhabung 8
Abbildung 5: Testplan (Beispiel) 10
Abbildung 6: Fehlermanagement im Testprozess 12
Abbildung 7: Fehlerkategorien 13
Abbildung 8: Aufbau einer Fehlergesamtliste 13
Abbildung 9: Bugzilla Benutzeroberfläche 14
Abbildung 10: Fehlerkommunikation und Fehlerbearbeitung 15
Abbildung 11: Beispiel für ein Statustracking (Testfortschritt) 16
Abbildung 12: Statusreporting – Fehlerstatistiken (exemplarisch) 17

Literatur / Links

[1] Wikipedia Enzyklopädie: http://de.wikipedia.org/wiki/Black-Box-Test

[2] Vigenschow, Uwe: „Trügerische Sicherheit – Unit-Tests richtig angewendet; Teil 2: eine minimale Testmethodik“ [oose.de, 2002]

[3] Schirmacher, Arne: „Testdokumentation nach ANSI/IEEE 829, Teil 1“, [„Die Software Testing Website“, 2001, http://www.softwaretesting.de/article/view/1/1/7/]

[4] Wilmer, Bernhard: „Testcaseverwaltung mit Excel - Teil2“, [„Die Software Testing Website“, 2001, http://www.softwaretesting.de/article/view/81/1/6/]

[5] Mozilla Foundation: Bugzilla Bugtracking Tool [http://www.bugzilla.org]

Abstract

Der nachfolgende Artikel beschreibt die Erfahrungen aus einem Anwendertest (funktionaler Black Box Test) im Rahmen eines Projektes im Customer Relationship Management (CRM) / Call Center Umfeld. Aus den hierbei gemachten Erfahrungen werden pragmatische Ansätze und Empfehlungen zur erfolgreichen Umsetzung der wesentlichen Schritte eines Anwendertests abgeleitet und anschaulich mit Empfehlungen, Vorlagen und zahlreichen grafischen Darstellungen beschrieben. Im Einzelnen wird der Testprozess entlang der folgenden Schritte dargestellt:

1. Die Phase der Testvorbereitung: konzeptionelle Vorbereitung, Aufbau der Testorganisation, Definition von Testfällen und Testskripten sowie Bereitstellung der Testumgebung und der erforderlichen Testdaten.
2. Darstellung der Vorgehensweise bei der Definition und Erstellung von Testskripten mit Beispielvorlagen und detaillierter Beschreibung ihrer Funktion.
3. Ansätze und Vorlagen für die Testplanung: Beschreibung einer möglichen Vorgehensweise für die Ressourcen- und Ausführungsplanung der Testzyklen
4. Die Phase der Testdurchführung: Verwendung der Testskripte im Testbetrieb, Darstellung des Fehlermanagementprozesses und Empfehlung geeigneter Tools für eine effiziente Fehlerverwaltung und -bearbeitung.
5. Beschreibung der Rolle der Testleitung in der Vorbereitungs- und Durchführungsphase.
6. Ansätze und Beispiele für ein Statuscontrolling und -reporting im Testbetrieb.
7. Empfehlungen für die Testdokumentation nach dem IEEE/ANSI 829 Standard.

Die hier beschriebenen Ansätze und Empfehlungen sowie die vorgestellten Vorlagen und Tools wurden zwar dem diesem Bericht zugrunde liegenden Anwendertest entnommen, jedoch weitestgehend von der konkreten fachlichen Testsituation abstrahiert, so dass sie auch auf beliebige andere Testsituationen übertragbar sind. Der diesem Bereicht zugrunde liegende Anwendertest wurde über einen Zeitraum von sieben Monaten als Abschlussphase eines mehrjährigen Systemeinführungs-, bzw. Optimierungsprojektes mit einem Aufwand von mehr als 400 Personentagen erfolgreich durchgeführt. Im Rahmen der Projektorganisation waren dem Testteam zwei Begleiter der ausführenden Fachbereiche, ca. 20 Tester, drei Entwicklungsteams und zwei externe Lieferanten Rollen zugeordnet. Der Autor dieses Berichts war der verantwortliche Testmanager.

1 Methodische Betrachtungen zum Testansatz und Begriffsbestimmungen

Bei dem im Folgenden beschriebenen funktionalen Test (Anwendertest) handelt es sich um einen so genannten *Black-Box-Test*. Dies bezeichnet eine Methode des Testens bei der die Tests ohne detaillierte Kenntnisse über die innere Funktionsweise des zu testenden Systems entwickelt und ausgeführt werden. Der Black-Box-Test beschränkt sich auf funktionsorientiertes Testen, d.h. für die Definition der Testfälle wird nur die fachliche Spezifikation (gewünschte Wirkung) in Form eines Fach- oder Grobkonzepts, nicht aber die spezifische Art der Implementierung des Testobjekts herangezogen. Das Ziel ist, die Übereinstimmung eines Softwaresystems mit seiner Spezifikation zu überprüfen (vgl. [1]).

Ausgehend von der formalen fachlichen Spezifikation werden Testfälle erarbeitet, welche sicherstellen sollen, dass der geforderte Funktionsumfang eingehalten wird. Das zu testende System wird dabei als abgeschlossenes Ganzes („Black Box“) betrachtet, bei der Bewertung der Testergebnisse wird nur sein Außenverhalten herangezogen.
Hierbei gilt es zu verhindern, dass z.B. in frühen Phasen des Softwareentwurfs erstellte Spezifikationen spätere Detail- und Implementierungsentscheidungen nicht vollständig abdecken. Abhilfe kann hier ein entsprechend „rigoroses“ Releasemanagement schaffen, in dem jede nachträgliche funktionale Änderung in der Spezifikation ebenfalls nachspezifiziert werden muss und somit im Anwendertest auch überprüft werden kann.

Dennoch muss einschränkend festgehalten werden dass auch ein erfolgreich abgeschlossener Black-Box-Test oftmals keine Garantie für grundsätzliche Fehlerfreiheit der Software gewährleisten kann, da ein vollständiger Test, der sämtliche denkbaren Funktionalitäten abdeckt, in der Regel wenig wirtschaftlich oder nicht realisierbar ist, da das Ableiten von Testfällen aus einer Spezifikation vergleichsweise aufwendig und je nach Präzisionsgrad der Spezifikation meist vollständig gar nicht möglich ist.
Ein Softwaretest selbst kann unter zwei unterschiedlichen Blickwinkeln betrachtet werden. Einerseits kann man das Testen von Software unter dem Aspekt des *Demonstrativen Testens* verstehen, mit dem Ziel die Qualität der Software zu erhöhen, indem definierte Funktionalitäten von Softwaresystemen unter sorgfältig kontrollierten Umständen systematisch ausgeführt werden. Andererseits kann man Testen auch als *Destruktives Testen* verstehen, was den Prozess der Ausführung von Software mit der Intention Fehler zu finden beschreibt (vgl. [2]). In diesem Sinne entspricht der hier gewählte Testansatz dem Demonstrativen Test.

Zur methodischen Abgrenzung sei noch der Begriff des *White-Box-Tests* erwähnt. Dieser bezeichnet eine Methode des Testens, bei der Tests mit Kenntnissen über die innere Funktionsweise des zu testenden Systems durchgeführt werden. Die Tests erfolgen hierbei anhand des Programmcodes mit dem Ziel spezifische, im Voraus bekannte Fehlermöglichkeiten auszuschließen. Diese Vorgehensweise findet in der Regel in den Phasen der Komponenten- und Integrationstests, welche dem Anwendertest vorausgehen, Anwendung.

2 Unterscheidung der Testphasen

Komplementär zu den Phasen der Softwareentwicklung, welche sich vereinfach in eine Grobkonzeptphase (fachliche Spezifikation), eine Feinkonzeptphase (technische Spezifikation) und die Realisierung (Anwendungsentwicklung) einteilen lassen, existieren entsprechende Testphasen, welche jeweils die Umsetzung der spezifischen Anforderungen systematisch überprüfen:

- *Komponententest (component test)*
  Test einzelner Funktionsbausteine innerhalb der Gesamtanwendung. Dieser Test wird nach Abschluss der Entwicklung einzelner Anwendungen oder Komponenten von Anwendungen durchgeführt und bildet die Voraussetzung für den nächsten Teilschritt, der Integration aller Komponenten. Grundlage für den Komponententest ist die technische Spezifikation der einzelnen Komponenten.
- *Integrationstest (integration test)*
  Test des Zusammenspiels von Funktionsbausteinen und Schnittstellen zur Sicherstellung der technischen Stabilität des Gesamtsystems sowie der korrekten Durchführung der Datenflüsse. Dieser Test wird nach Abschluss der Integration aller Komponenten durchgeführt. Grundlage für den Integrationstest ist die technische Spezifikation des Gesamtsystems.

- *Anwendertest (user acceptance test)*
  Test der Gesamtanwendung zur ganzheitlichen Überprüfung der Erfüllung der fachlichen Anforderungen im Rahmen einer simulierten Zielorganisation. Dieser Test wird nach Abschluss des Integrationstests durchgeführt. Die Grundlage für den Anwendertest bildet das Fachkonzept (Spezifikation der fachlichen Anforderungen an das Gesamtsystem, oft auch als *Grobkonzept* oder *Pflichtenheft* bezeichnet).

Da sowohl die hier „Integrationstest“ genannte Testphase als auch der „Anwendertest“ sich auf einen Test der integrierten Gesamtlösung beziehen und den Fokus auf jeweils unterschiedliche Aspekte bezüglich der Testinhalte (technischer und funktionaler/fachlicher Test) sind in der Praxis hierfür gelegentlich auch die Begriffe *„technischer Integrationstest“* sowie *„fachlicher* bzw. *funktionaler Integrationstest“* gebräuchlich. Eine Übersicht über die Entwicklungsphasen und die hierzu komplementären Testphasen zeigt die folgende Abbildung:

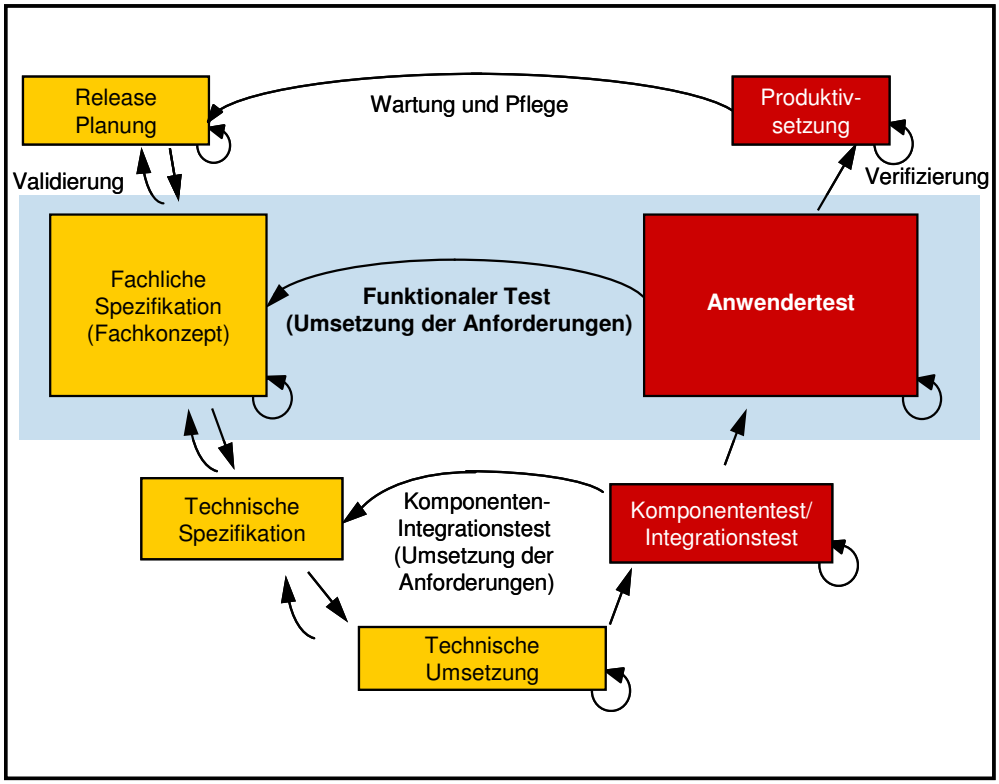


Abbildung 1: Schematische Darstellung der Testphasen

Es wird empfohlen, nach jeder Testphase ein Meilenstein Review durchzuführen, in dem die wesentlichen Ergebnisse einer Phase vor einem Expertengremium präsentiert werden. Das Review ist ein standardisierter Freigabeprozess, bei dem Anhand einer standardisierten Checkliste über die Freigabe einer Anwendung (z.B. für die nächste Testphase) entschieden wird. Die Checkliste mit spezifischen Fragen zur „Reife“ der Anwendung wird in der Review Sitzung gemeinsam durchgegangen. Das Ergebnis des Reviews ist ein Protokoll, in dem die wesentlichen Ergebnisse festgehalten werden. Das Endergebnis kann sein:

- Eine Abnahme ohne Auflagen wird erteilt (keine Einschränkungen)
- Eine Abnahme unter Auflagen wird erteilt (Freigabe verzögert)
- Eine Abnahme ist nicht möglich (Freigabe verhindert)

Falls keine freigabeverhindernden Punkte identifiziert wurden, kann nach Planung/Beseitigung der freigabeverzögernden Punkte die Freigabe erteilt werden. Die Behandlung aufgetretener freigabeverhindernder Punkte wird im Einzelfall entschieden.

3 Vorbereitende Schritte

3.1 Testvorbereitung und Testorganisation

Folgende Voraussetzungen müssen im Rahmen der Testvorbereitung erfüllt werden:

- Benennung einer *Testorganisation*, vertreten durch die Testleitung, welche ein Testteam zusammenstellt und die Erstellung und Abnahme des *Testmodells* steuert. Das Testmodell selbst beinhaltet folgende Komponenten:
  - Definition der *Testzyklen* und *Testbedingungen*, der *Testskripte*, welche die erwarteten Ergebnisse enthalten, sowie der *Testdaten* zur Durchführung der Testfälle / Testskripte.
  - Definition und Abstimmung des *Prozesses zur Testdurchführung, Fehlerbehebung und Freigabe der Anwendung* mit den verantwortlichen Stellen (Auftraggeber, Projektleitung, Fachbereichsverantwortliche), sowie der *Rollen und Verantwortlichkeiten im Testprozess*.
- Bereitstellung der *Testinfrastruktur*. Das Testsystem (Testumgebung) wird als Abbild bzw. Ausschnitt des Produktivsystems eingerichtet und sollte zumindest die zur Bereitstellung der zu testenden Funktionalitäten erforderlichen Komponenten beinhalten. Um unerwünschte Wechselwirkungen zu vermeiden ist darauf zu achten, dass es keinerlei Verbindung zwischen Test- und Produktivsystem gibt. Darüber hinaus sind folgende Maßnahmen sicherzustellen:
  - Die zu testenden Anwendungskomponenten sind auf der Testumgebung bereitgestellt
  - Alle erforderlichen User IDs und Berechtigungen für Testressourcen sind eingerichtet
  - Die Test PCs sind vollständig mit allen erforderlichen Softwarekomponenten aufgesetzt
  - Die erforderliche Netzwerk-Konfiguration (Router, Ports, Firewall,...) ist durchgeführt
  - Es ist sichergestellt, dass während des laufenden Tests keine Änderungen an der technischen Infrastruktur vorgenommen werden (Stabilität)

Die zuvor genannte Punkte, die *Testorganisation*, das *Testmodell*, die *Testinhalte*, der *Testprozess* sowie die *Testinfrastruktur* sollten im Rahmen eines *Testkonzepts* detailliert beschrieben werden (vgl. hierzu Abschnitt 6.1.1). Ferner empfiehlt sich eine Abstimmung und Freigabe des Testkonzepts, durch die betroffenen Bereiche im Rahmen eines formalen Abnahmeprozesses.

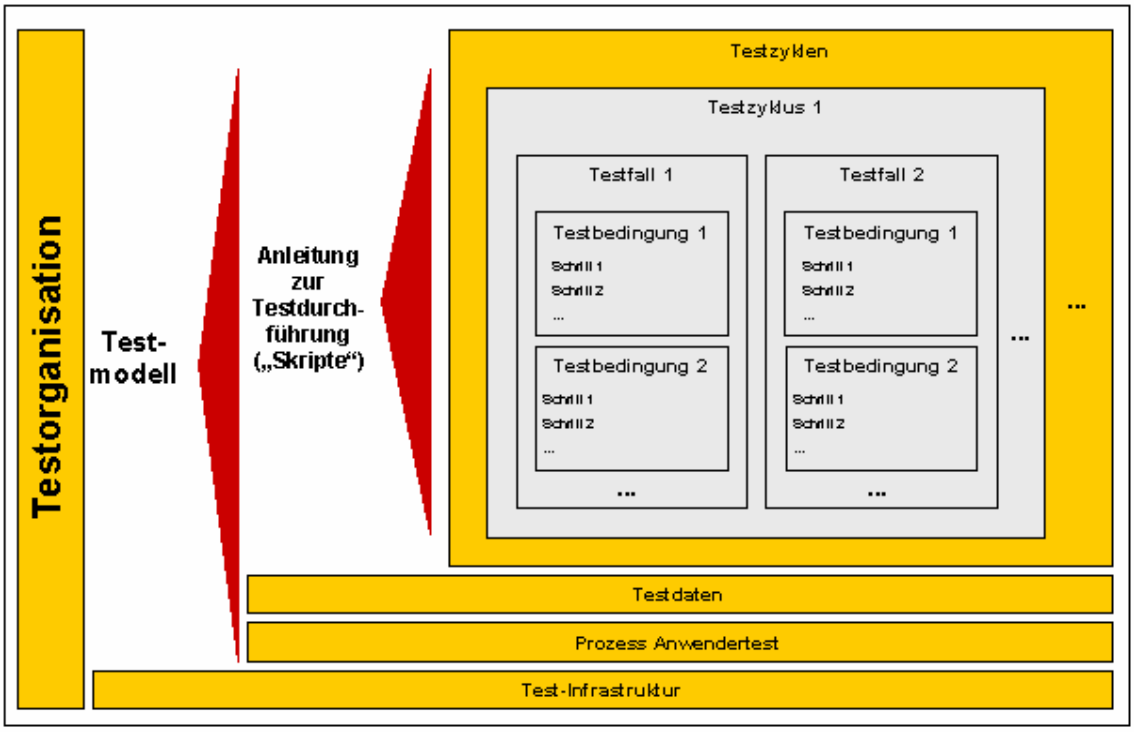


Abbildung 2: Testmodell und Testorganisation im Anwendertest

3.1.1 Definition von Testfällen und Testskripten

Da der funktionale Test oder Anwendertest seinem Wesen nach, wie bereits dargestellt, ein klassischer *Black-Box-Test* ist, erfolgt die Konzeption und Vorbereitung des Tests auf Basis des Grob- oder Fachkonzepts (auch: funktionale Spezifikation). Hierfür werden in einem ersten vorbereitenden Schritt auf Grundlage der im Fachkonzept definierten fachlichen Anforderungen an das Gesamtsystem und seine Teilkomponenten *Testskripte* und *Testfälle* abgeleitet.
Diese wiederum bilden eine detaillierte, schrittweise Beschreibung für eine Überprüfung des korrekten Systemverhaltens bezüglich der im Fachkonzept definierten Geschäftsvorfälle. Im Zuge der Erstellung der Testskripte wird empfohlen zunächst in einem Top Down Ansatz die zu testenden Funktionalitäten zu logischen Funktionsblöcken zusammen zu fassen. Hieraus können anschließend detaillierte Anweisungen abgeleitet werden, welche Schritte (Aktionen) ein Benutzer (Tester) auszuführen hat, und welches Systemverhalten hieraus resultieren soll. Ein Testskript umfasst einen logisch in sich zusammenhängenden und abgeschlossenen Vorgang mit definierten Eingangs- und Ausgangsbedingungen, welcher einen bestimmten Vorgang im Produktivbetrieb simuliert.

Werden innerhalb eines Testskripts unterschiedliche bzw. alternative Ausprägungen eines bestimmten Vorgangs oder Geschäftsvorfalls behandelt, so spricht man von verschiedenen Testfällen innerhalb des Testskripts. In der Praxis werden die Begriffe *Testfall* und *Testskript* häufig synonym verwendet. Im Folgenden soll jedoch gelten, dass ein Testfall eine Kombination von Eingabedaten, Bedingungen und erwarteten Ausgaben darstellt. Ein Testfall enthält keine Angaben zur Bedienung oder Ausführung des Testfalls. Diese findet man im dazugehörigen Testskript, auf das ein Testfall verweist oder das diesen Testfall als Teilmenge enthält (vgl. [3]). Nachfolgend sind beispielhaft einige Funktionsblöcke und die daraus abgeleiteten Testskripte aus dem hier zugrunde liegenden Fall (Test aus dem CRM / Call Center Bereich) dargestellt:

Abbildung 3: Übersicht der definierten Testskripte

3.1.2 Erstellung und Verwendung von Testskripten

Für die Erstellung und Verwaltung der Testskripte bietet Microsoft Excel® einige Vorteile. Es bietet ohne großen Aufwand ein einheitliches Layout für Testskripte, welches in allen Testabschnitten verwendet werden kann und die Tester können ohne Formatierungsprobleme die Skripte während der Testausführung direkt am Bildschirm bearbeiten. Außerdem können Testskriptvorlagen auf der Basis von Microsoft Excel® bei Wiederholungen von Tests beliebig oft wieder verwendet werden.
Ein weiterer Vorteil von Microsoft Excel® liegt darin, dass in einer Testskriptdatei neben dem eigentlichen Testskript, welches auf einem Tabellenblatt schrittweise die detaillierten Handlungsanweisungen für die Testperson sowie das erwartete und hieraus resultierende Systemverhalten beschreibt, auf weiteren Tabellenblättern auch die für den Test erforderlichen Testdaten enthalten sein können.

Werden innerhalb eines Testskripts unterschiedliche Alternativen eines Vorgangs überprüft (unterschiedliche Testfälle), so können diese ebenfalls auf weiteren Tabellenblättern innerhalb des gleichen Testskripts beschrieben werden. Schließlich ermöglicht es auch noch einfache mathematisch-statistische Auswertungen direkt nach Abschluss der Tests innerhalb der jeweiligen Testskripte (z.B. Fehlerhäufigkeiten). Ein Beispiel für den Aufbau einer Testskriptvorlage zeigt die folgende Abbildung:

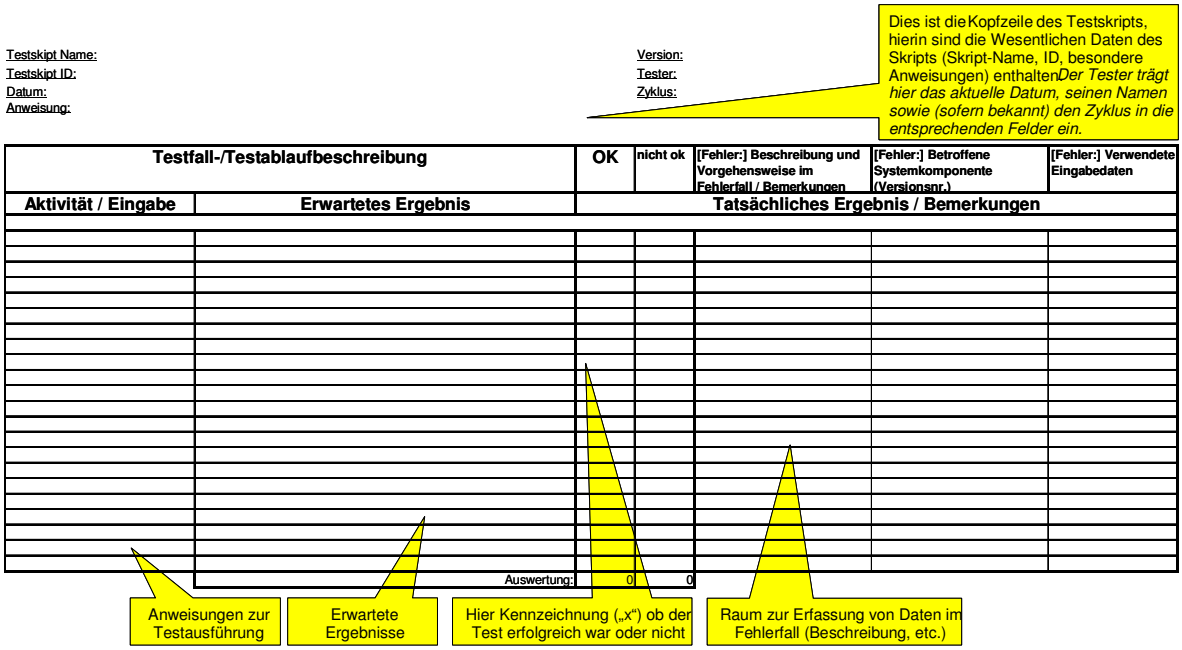


Abbildung 4: Testskript – Beispiel für Aufbau und Handhabung

Das Testskript enthält einen Kopfzeilenbereich, in welchem der Name des Testskripts sowie eine eindeutige Testskript ID (vgl. Abschnitt 3.1.3) eingetragen werden. Darüber hinaus können hier weitere Daten zur Testskriptverwaltung und späteren Auswertung erhoben werden. In der Regel werden hierfür zumindest das Datum der Testausführung und der Name des Testers eingetragen. Weiterhin können je nach spezifischer Anforderung auch Angaben zur getesteten System- bzw. Komponentenversion, zum verwendeten Login oder zum aktuellen Testzyklus gemacht werden, sowie allgemeine Anweisungen zur Testausführung beschrieben werden.

Das Testskript stellt in übersichtlicher Art und Weise die vom Tester auszuführenden Aktivitäten und Eingaben sowie die erwarteten Ergebnisse dar. Daneben hat der Tester die Möglichkeit, die tatsächlichen Ergebnisse festzuhalten. Dieses erfolgt hier durch Setzen von Kreuzen („X“) in die entsprechenden Felder der Spalten mit den Überschriften „OK“ bzw. „nicht OK“ bezogen auf eine bestimmte Aktivität/Eingabe und das hieraus resultierende Ergebnis. Schließlich folgen für den Fall eines Abweichens des tatsächlichen Systemverhaltens vom erwarteten Systemverhalten („Fehlerfall“) Felder zur detaillierten Beschreibung des Fehlers, sowie je nach Bedarf für ergänzende Angaben wie z.B. die betroffene Systemkomponente oder die bei der Erzeugung des Fehlerfalls verwendeten Eingabedaten.

Testskripte werden zentral von der Testleitung verwaltet und den Testern täglich zur Bearbeitung (Testausführung) zugewiesen. Nach Abschluss eines Tests – entweder erfolgreich nach vollständigem Durchlaufen des Testskripts, oder nach Abbruch bei Auftreten von (schweren) Fehlern[1] – wird das Testskript ausgedruckt und vom Tester unterschrieben an die Testleitung zur Auswertung, Ablage und späteren Archivierung ausgehändigt.

[1] Je nach zugrunde liegender Testsystematik und den spezifischen Eigenschaften des zu testenden Systems muss im Einzelfall entschieden werden, ob jeder in einem Test auftretende Fehler automatisch zu einem Abbruch des Tests führt, oder ob trotz Fehlern der Test bis zum Ende auszuführen ist.

Die Dokumentationsfunktion der bearbeiteten und unterzeichneten Testskripte ist vor allem im späteren Produktivbetrieb, nach Abschluss der Tests, von Bedeutung, um im Zweifelsfalle nachweisen zu können, dass bestimmte Funktionalitäten, welche trotz erfolgreicher Tests ggf. beanstandet werden können, entsprechend der vereinbarten und mit den beteiligten Parteien (z.B. Auftraggeber) abgestimmten Testdefinition auch ausgeführt wurden. Auch kann hierüber die Historie von Systemfehlern und ihre Behebung nachvollzogen werden.

3.1.3 Verwaltung der Testskripte und Vorbereitung der Testplanung

In größeren Tests kann die Menge der auszuführenden Testskripte schnell sehr groß[2] werden. Von daher ist für ein effizientes Testmanagement eine effektive Verwaltung der Testskripte erforderlich. Im Folgenden wird ein beispielhafter Ansatz hierfür beschrieben.
Zunächst werden geeignete Funktionsblöcke identifiziert, denen die Testskripte zugeordnet werden können (z.B. im hier vorliegenden Fall die Funktionsblöcke *Inbound, Outbound, Reporting*). Die Funktionsblöcke werden anschließend weiterhin in funktionale Teilaspekte aufgegliedert, durchnummeriert und anschließend mit einer laufenden Testskriptnummer versehen, wodurch sich folgende Systematik für eine Testskript ID ergibt:

1.Stelle Testskript ID
X: Testabschnitt (z.B. hier I für *Inbound*)
2. Stelle Testskript ID:
01 Funktionsblock 1
02 Funktionsblock 2
03 Funktionsblock 3 (…)
3. Stelle Testskript ID:
Laufende Testskript Nr. (hier 5-Stellig, kann jedoch beliebig gewählt werden)

Die Testskript ID ist auch bei der Bennennung der Testskriptdokumente stets mit einzufügen. Ein Beispielhafter Dateiname lautet demnach *„I-02-00050 Erkennungsgenauigkeit Spracheingabe.xls“*. Dies bedeutet, dass es sich um ein Testskript aus dem Testabschnitt *Inbound* handelt, hierin gehört es zu den Tests der *Erkennungsgenauigkeit* und hat die laufende Nummer 50. Die Testskriptnummern sollten global und fortlaufend vergeben werden, nicht nur innerhalb der einzelnen Testabschnitte. Insbesondere sollten auch Testskripte, welche z.B. nachträglich in einen Bestimmten Testabschnitt eingefügt werden, zur Vermeidung einer Doppelnummerierung weiterhin fortlaufende Nummern erhalten.

Eine durchgehende Systematik, wie hier beispielhaft dargestellt, kann und sollte den jeweiligen Gegebenheiten des zu testenden Systems angepasst werden. Anhand der Testskript ID kann nun jedes Testskript in der Planung und Durchführung des Tests sowie in der Fehlerkommunikation und im Statusreporting stets eindeutig identifiziert und zugeordnet werden.

Für den nächsten Schritt – die Testplanung – wird eine Übersichtsliste mit allen auszuführenden Testskripten, gegliedert z.B. nach fachlichen Aspekten angelegt (vgl. Abbildung 5). Diese Liste (*„Execution Plan“*) ist ein zentrales Tool für die Ausführungsplanung der einzelnen Tests sowie für die Statusverfolgung auf Testskriptebene. Sie enthält neben der Auflistung aller auszuführenden Testskripte gegliedert nach Funktionsblöcken auch den für jedes Testskript geschätzten Zeitaufwand (vgl. Abschnitt 4.1). Daneben sollten für die Anzahl der vorgesehenen Testzyklen (in der Regel ist von 4-5 Testzyklen bis zur Fehlerfreiheit auszugehen) Spalten für den Tag der geplanten Testausführung, das Ergebnis je Testskript und die vorgesehenen Tester enthalten sein. Bei Bedarf kann die Ausführungsplanung sowie die jeweiligen Ergebnisse durch Verknüpfung der jeweiligen Zellen in einer Übersicht visualisiert werden.

Es wird empfohlen im Prozess der Testplanung sowie auch bei der Definition der Testskripte frühzeitig die beteiligten Fachbereiche und – sofern möglich – auch die vorgesehenen Tester mit einzubinden um die Qualität und Akzeptanz der Ergebnisse zu erhöhen.

---

[2] Der diesem Bericht zugrunde liegende Anwendertest umfasste insgesamt die Ausführung von 199 Testskripten in drei verschiedenen, aufeinander folgenden Testphasen (Inbound, Outbound und Reporting).

4 Testplanung, Ausführung und Statuscontrolling

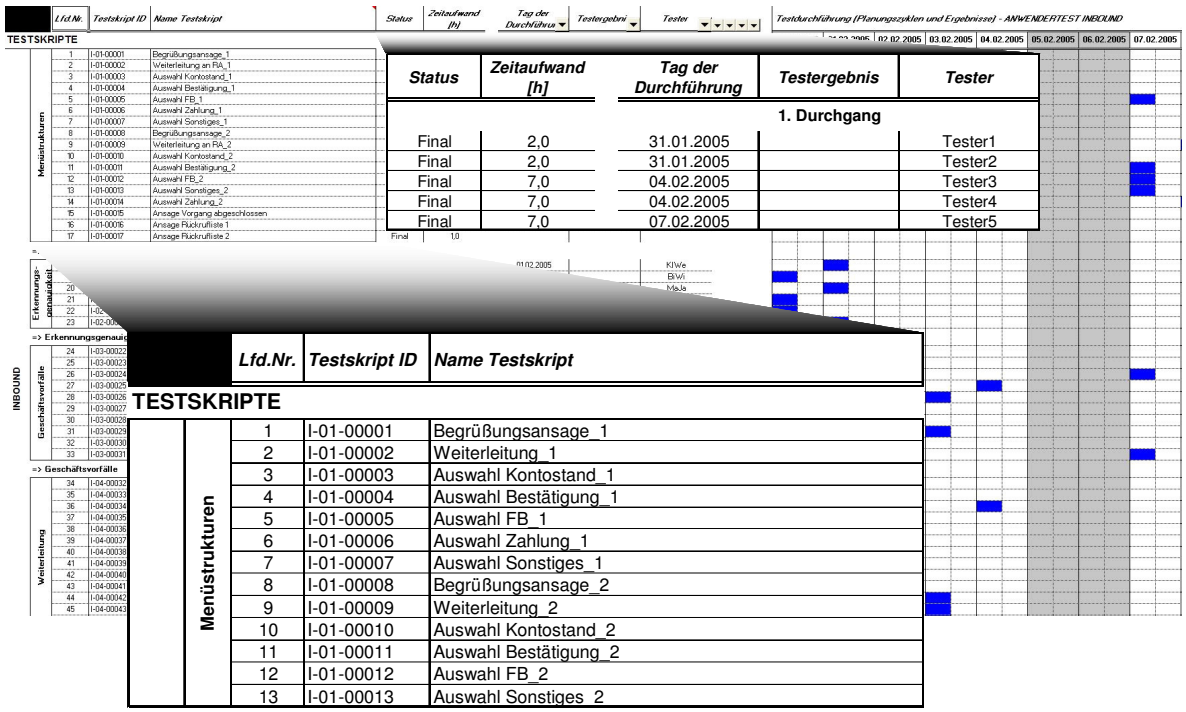
4.1 Testplanung

Für die Testplanung werden in einem ersten Schritt die Zeitaufwände für die Ausführung jedes einzelnen Testskripts geschätzt. Dies kann im Rahmen von Planungs-/Schätzworkshops mit den beteiligten Bereichen (z.B. mit den Verantwortlichen für die Erstellung der fachlichen Spezifikation) geschehen. Die geschätzten Zeiten werden in die zuvor erstellte Übersichtsliste der auszuführenden Testskripte eingetragen.

Voraussetzung für die Durchführung eines Anwendertests ist die Verfügbarkeit von Testressourcen (Testern) zur Ausführung der Testskripte. Testressourcen werden zumeist von den Bereichen, welche einen Anwendertest, bzw. das Entwicklungsprojekt, in das ein Anwendertest eingebettet ist, in Auftrag gegeben haben, bereitgestellt. In der Regel jedoch nicht zu 100% als VZE[3], sondern oftmals nur zu 50% bzw. 25% (d.h. für 4h bzw. 2h je Tag). Vor dem Hintergrund der Verfügbarkeit der Testressourcen (Ressourcenplan) können den einzelnen Testressourcen nun je Tag Testskripte entsprechend der für ihre Ausführung veranschlagten Zeiten zugewiesen werden, bis die maximale tägliche Ressourcenauslastung erreicht ist.

Es empfiehlt sich zu Beginn eine Sicherheitsreserve je Testressource von 10-20% einzuplanen z.B. für Einlernzeiten sowie für die vor allem zu Beginn deutlich erhöhte Betreuungsintensität. Im konkreten Beispiel sollten daher einem Tester, der für 4h je Tag zur Verfügung steht, nur Testskripte zugewiesen werden, deren Bearbeitung in Summe ca. 3 - 3,5h ausmacht. Die Restliche Zeit dient als Reserve bzw. Puffer. Mit fortschreitendem Testverlauf und zunehmender Erfahrung der Tester kann der Puffer allmählich abgebaut und die effektive Ressourcenauslastung somit erhöht werden. Auf diese Weise werden die auszuführenden Testskripte den verfügbaren Testressourcen jeweils auf Tagesbasis zugewiesen und somit eine verbindliche Testplanung für den *ersten Testzyklus* erstellt.

Ein Beispiel für einen derartigen Testplan (Execution Plan) ist in der folgenden Abbildung gezeigt:



Lfd.Nr.	Testskript ID	Name Testskript	Status	Zeitaufwand [h]	Tag der Durchführung	Testergebnis	Tester
		TESTSKRIPTE					
					1. Durchgang		
1	I-01-00001	Begrüßungsansage_1	Final	2,0	31.01.2005		Tester1
2	I-01-00002	Weiterleitung an RA_1	Final	2,0	31.01.2005		Tester2
3	I-01-00003	Auswahl Kontostand_1	Final	7,0	04.02.2005		Tester3
4	I-01-00004	Auswahl Bestätigung_1	Final	7,0	04.02.2005		Tester4
5	I-01-00005	Auswahl FB_1	Final	7,0	07.02.2005		Tester5
6	I-01-00006	Auswahl Zahlung_1					
7	I-01-00007	Auswahl Sonstiges_1					
8	I-01-00008	Begrüßungsansage_2					
9	I-01-00009	Weiterleitung_2					
10	I-01-00010	Auswahl Kontostand_2					
11	I-01-00011	Auswahl Bestätigung_2					
12	I-01-00012	Auswahl FB_2					
13	I-01-00013	Auswahl Sonstiges_2					

Abbildung 5: Testplan (Beispiel)

[3] VZE = Vollzeiteinheit (entspricht einer Person, die 8h je Tag zur Verfügung steht), oft ist hierfür auch die Englische Abkürzung *FTE (Full Time Equivalent)* gebräuchlich.

Eine Planung der nachfolgenden Zyklen ist zu Beginn des Anwendertests nicht sinnvoll, da erfolgreich durchlaufene Testskripte nicht wiederholt werden müssen und somit die Anzahl der auszuführenden Testskripte in den Folgezyklen von den Ergebnissen der vorangegangenen Zyklen abhängt. In den Folgezyklen werden auf die gleiche Art und Weise diejenigen Testskripte eingeplant, für die sämtliche in den vorangegangenen Zyklen aufgetreten Fehler als behoben gemeldet wurden. Dies geschieht jeweils kurzfristig, sobald die Testskripte für einen Nachtest zur Verfügung stehen.

4.2 Testausführung

Die eigentliche Testausführung in einem Anwendertest besteht aus der Ausführung der jeweiligen Testskripte und der Dokumentation der Ergebnisse. Hierzu werden die in den Testskripten beschriebenen Anweisungen schrittweise durchlaufen. Dabei wird für jeden Anweisungsschritt im Testskript festgehalten, ob das tatsächliche Ergebnis dem erwarteten Ergebnis entspricht (fehlerfreie Ausführung) oder ob es hiervon Abweichungen gibt. In diesem Fall spricht man von einem Fehler.
Die Tester dokumentieren auftretende Fehler direkt in den Testskripten. Hierzu sind eine kurze Beschreibung des Fehler, sowie der Aktionen, die zu seinem Auftreten führten, erforderlich. Darüber hinaus können weitere Angaben zur konkreteren Bestimmung und Klassifizierung des Fehlers zur Unterstützung des Fehlermanagements hilfreich sein, wie z.B. die betroffene Systemkomponente, Datum und Uhrzeit des Fehlers oder die bei Auftreten des Fehlers verwendeten Eingabedaten. Hierfür sind in den Testskripten gesonderte Felder vorgesehen (vgl. Abbildung 4).
Neben den eigentlichen Testressourcen (Testern) sind an der Testausführung in der Regel auch Vertreter der Fachbereiche zur Klärung von im Testverlauf auftretenden fachlichen Fragen bzw. Unklarheiten, Administratoren zur Überwachung des korrekten System- und Testbetriebs sowie Mitarbeiter der Anwendungsentwicklung zur Entgegennahme und Bearbeitung von gemeldeten Fehlern beteiligt.

4.2.1 Fehlermanagement

Ziel eines Anwendertests ist es, sämtliche Fehler einer zu testenden Software, bzw. eines zu testenden Systems im Testverlauf zu finden und bis zur Produktivsetzung der Software oder des Systems zu eliminieren, wobei als Fehler jegliche Abweichung des tatsächlichen Systemverhaltens von seiner fachlichen (funktionalen) Spezifikation gewertet wird. Um dieses Ziel im vorgegebenen, oft sehr eng gesteckten, Zeitrahmen zu erreichen ist ein effektives Fehlermanagement erforderlich. Der folgende Ansatz hat sich hierfür bewährt:

- Die Tester tragen im Testverlauf aufgetretene Fehler direkt im jeweiligen Testskript ein und übergeben dieses nach Abschluss der Bearbeitung an die Testleitung (oder eine andere für das Fehlermanagement verantwortliche Stelle).
- Jeder Fehler erhält bei Erfassung eine eindeutige Bezeichnung (Fehlernummer) mit einer Referenz zum zugrunde liegenden Testskript, in dessen Kontext er auftrat, sowie eine Beschreibung
- Die Fehlerbeschreibung sollte neben der detaillierten Beschreibung des Problems Angaben zum Tester, zu Datum und Uhrzeit der Testausführung, zur verwendeten Testumgebung / Softwareversion, auch die Aktionen enthalten, welche zu seinem Auftreten führen, sowie betroffene Systemkomponenten und verwendete Eingabedaten. Idealerweise können diese Daten direkt dem Testskript entnommen werden. Ist dies nicht der Fall, so ist im Zweifelsfalle der Tester, der den Fehler bemerkte hierzu zu befragen.
- Die Testleitung klassifiziert und priorisiert auftretende Fehler nach ihrer Schwere (vgl. Abbildung 7) und leitet diese an die für die Fehlerbehebung zuständigen Stellen weiter.
- Testskripte mit Fehlern werden im Testbetrieb zurückgestellt und erst nach erfolgreicher Fehlerbehebung erneut in den Testlauf eingespielt (Nachtest).

Das folgende Bild beschreibt den Prozess der Testausführung und das Fehlermanagement. Im Fehlerfall trägt der Tester den gefundenen Fehler und alle hierfür erforderlichen Daten in das Testskript ein. Nach Ausführung des Tests reicht er das ausgefüllte und unterschriebene Testskript an die Testleitung bzw. eine andere für das Fehlermanagement verantwortliche Stelle weiter (ggf. ergänzt um zusätzliche Angaben, Screenshots etc.). Diese teilt den gemeldeten Fehlern eine eindeutige ID zu (z.B. eine laufende Fehlernummer) und leitet die Fehler an die Entwicklungsorganisation zur Bearbeitung weiter. Sind alle Fehler in einem Testskript behoben, kann dieses nachgetestet werden.

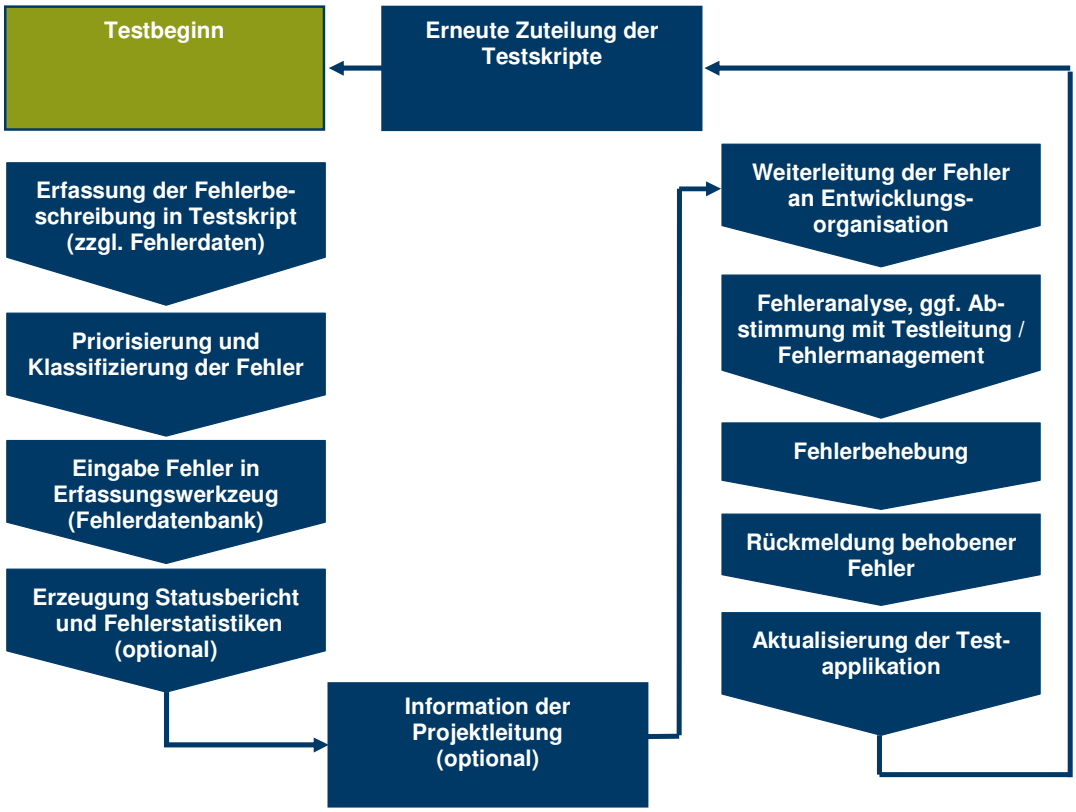


Abbildung 6: Fehlermanagement im Testprozess

4.2.1.1 Fehlerkategorien

Folgende Fehlerkategorien wurden im hier beschriebenen Fall zur Klassifizierung der identifizierten Fehler herangezogen, gemeinsam mit dem definierten Service Level zur Fehlerbehebung.

A	**Schwerer Fehler** Der Produktivstart ist gefährdet. Die Fortsetzung des Tests ist nicht möglich. **Die Behebung des Fehlers erfolgt sofort.** *Beispiel*: Die Eingabe von Daten ist nicht möglich. Berichte werden nicht ausgegeben. Eine Schnittstelle geht nicht.
B	**Mittelschwerer Fehler** Die Fortsetzung des Tests ist mit Hilfe eines Workaround (Zwischenlösung) möglich. **Die Behebung des Fehlers erfolgt spätestens innerhalb von einer Woche nach Produktivstart.** *Beispiel*: ein Pausenzustand ist nicht erreichbar; Zwischenlösung: Wahl eines anderen Pausenzustands
C	**Leichter Fehler** Die Durchführung des Tests ist nicht behindert. **Die Behebung des Fehlers muss *zwei* Wochen nach Produktivstart erfolgt sein.** *Beispiel*: Rechtschreibfehler in Dialogen

Die Testleitung kann bei Bedarf weitere Kategorien zur Klassifizierung verwenden, wie zum Beispiel:

VV	**Verbesserungsvorschlag**
NR	**Nicht reproduzierbar**
FT	**Fehlerhafte Testdefinition**
FD	**Fehlerhafte Testdaten**

Abbildung 7: Fehlerkategorien

4.2.1.2 Fehlererfassung und -kommunikation

Bei größeren Anwendertests kann die Anzahl der Fehler schnell sehr groß werden[4]. Von daher empfiehlt es sich, ein geeignetes Werkzeug zur Fehlererfassung, -kommunikation sowie zur Statusverfolgung zu verwenden. Hierfür existiert eine Anzahl von Tools, welche Teilweise als Freeware oder Open Source Software frei verfügbar sind. Auch können hier ebenfalls Office Tools wie Microsoft Excel® zum Einsatz kommen. Die einzelnen Tools haben jeweils spezifische Vor- und Nachteile, daher wird im Folgenden ein kombinierter Einsatz des Fehlererfassungswerkzeugs *Bugzilla* zur Fehlererfassung und Kommunikation mit den Entwicklungsteams, sowie Microsoft Excel® zum führen einer Fehlergesamtliste sowie zur Generierung entsprechender Statistiken und Fehlerreports für das Statusreporting empfohlen. Die folgende Grafik zeigt die Datenfelder, welche die Fehlergesamtliste enthalten sollte.

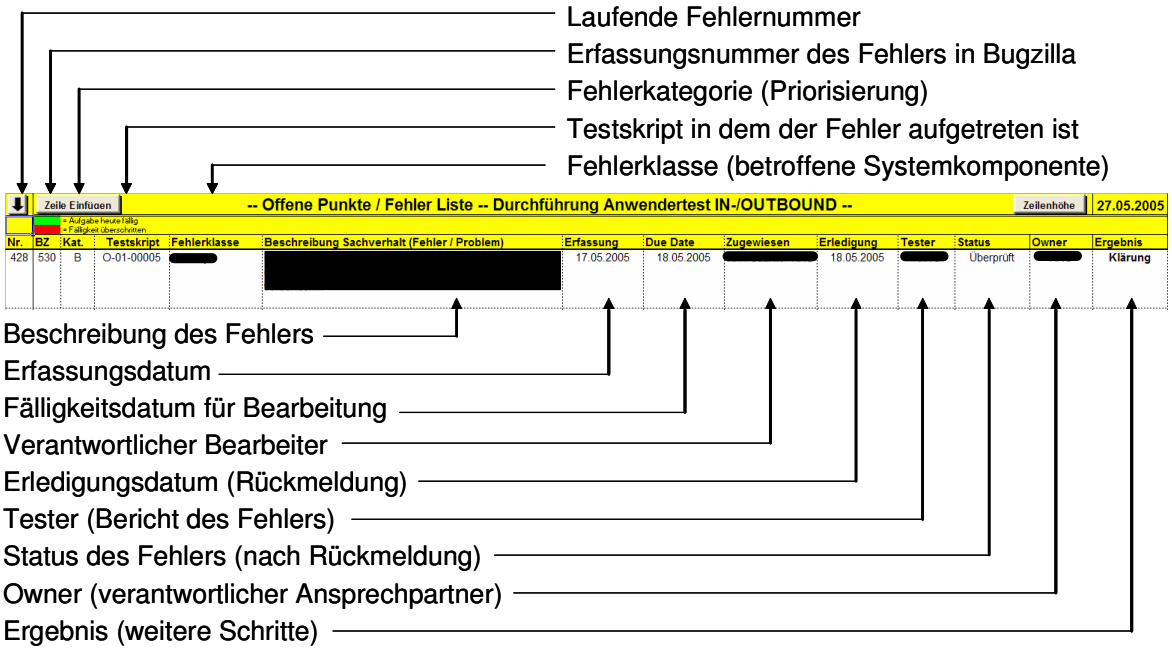


Abbildung 8: Aufbau einer Fehlergesamtliste

Bugzilla ein webbasiertes, frei erhältliches Fehlererfassungswerkzeug (Open Source) der Mozilla Foundation (vgl. [5]). Es handelt sich um ein CGI Programm, das heißt, dass es auf einem Webserver installiert wird und aus Anwendersicht wie eine Website funktioniert. Es ist daher nicht erforderlich, auf dem Arbeitsplatz eines Testers zusätzliche Software zu installieren.

[4] Im dem diesem Erfahrungsbericht zugrunde liegenden Fall wurden im Verlauf des Anwendertests insgesamt exakt 500 Fehler gefunden, dokumentiert, behoben und nachgetestet.

Die wesentlichen Merkmale dieser Software sind:

- Es können beliebig viele Benutzer, Projekte und Komponenten verwaltet werden
- Es können beliebig viele Kommentare zu einzelnen Einträgen hinzugefügt werden
- Einzelne Entwickler können bei bestimmten Fehlern gezielt adressiert werden
- Eine Bearbeitungshistorie hält jede Statusveränderung sowie den jeweiligen Bearbeiter fest.

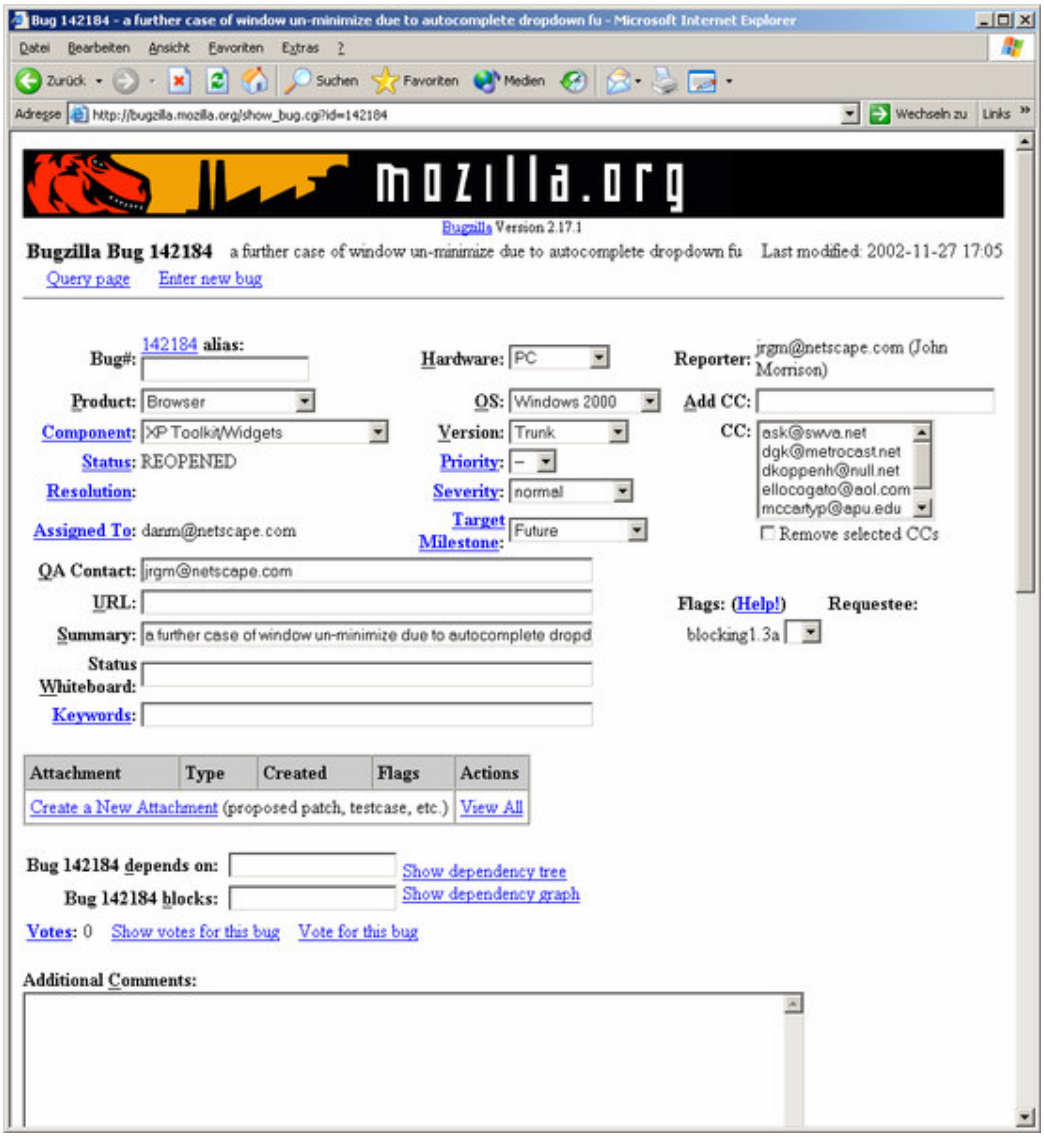


Abbildung 9: Bugzilla Benutzeroberfläche

Bugzilla unterstützt den Prozess der Fehlererfassung, -kommunikation und Rückmeldung der bearbeiteten Fehler durch einen definierten Workflow. Zu Beginn wird ein im Test aufgetretener und dokumentierter Fehler erfasst und einer bestimmten Software- bzw. Systemkomponente zugeordnet. Zugleich wird der Fehler auch einem bestimmten Bearbeiter zugewiesen. Erfolgt keine spezifische Zuweisung, so ist der Fehler automatisch dem zuvor definierten *owner* der Komponente zugewiesen.

Der Fehler hat jetzt den Status *new* und der zugewiesene Bearbeiter erhält eine automatisch generierte Email mit einer Kurzbeschreibung des Fehlers und einem Link zu dem entsprechenden Dokument in der Fehlerdatenbank. Der Bearbeiter kann dann entweder den Fehler selbst zur Bearbeitung annehmen oder einem anderen Bearbeiter zuweisen. Der Fehler erhält dadurch den Status *assigned*.

Ist der Fehler behoben oder wird die Bearbeitung abgelehnt (z.B. wenn nicht reproduzierbar), dann erhält der ursprüngliche Verfasser wiederum eine Email mit einem Link zu dem entsprechenden Dokument in der Fehlerdatenbank. Der Fehler hat dann z.B. den Status *fixed/resolved* oder *wontfix/invalid*. Erweist sich der Fehler schließlich im Nachtest als behoben, kann er vom ursprünglichen Verfasser geschlossen werden (Status *closed*). Generell erzeugt jede Veränderung eine Email so dass der Status jeder einzelnen Fehlermeldung von allen an einem spezifischen Fehler Beteiligten verfolgt werden kann.

Vorteile von Bugzilla liegen vor allem in der Unterstützung der Zusammenarbeit verteilter Teams, sowie in der Workflowunterstützung bei der Erfassung, Bearbeitung und Rückmeldung von Fehlern. Als Nachteil erwiesen sich insbesondere die nur eingeschränkten Statusverfolgungs- und Reportingmöglichkeiten. Insbesondere ermöglicht Bugzilla nur eine Fehlerstatusverfolgung auf Einzelfehlerebene, nicht jedoch auf Testskriptebene. Letzteres ist allerdings erforderlich, um sicherzustellen, dass Testskripte erst dann in den Nachtest einfließen, wenn sämtliche hierin aufgetretenen Fehler bearbeitet sind. Dies macht das Führen einer separaten, zentralen Fehlerliste auf Basis von in Microsoft Excel® erforderlich (vgl. Abbildung 8).

Organisatorisch empfiehlt es sich, den Prozess des Fehlermanagements bei einer verantwortlichen Stelle anzusiedeln, um Konsistenz im Fehlermanagement und jederzeitige Auskunftsfähigkeit (z.B. im Statusreporting) zu gewährleisten. Dies kann entweder die Testleitung selbst sein oder eine von der Testleitung im Rahmen der Testorganisation mit dieser Aufgabe beauftragte Person.
In der nachfolgend beispielhaft skizzierten Organisation für den Fehlerbearbeitungsprozess leitet die für das Fehlermanagement verantwortliche Stelle (hier: Testleitung) die Fehler nach Erfassung je nach betroffener Systemkomponente mit Hilfe von Bugzilla Fehlermeldungen an zentrale Koordinatoren weiter, welche diese innerhalb ihrer Bereiche bestimmten Entwicklern zur Bearbeitung zuweisen. Es ist auch möglich die Fehler direkt den entsprechenden Entwicklern (sofern diese bekannt sind) zuzuweisen. Die Rückmeldung der Bearbeitung von Fehlern erfolgt ebenfalls über Bugzilla Fehlerstatusmeldungen an die Testleitung, welche diese in einer zentralen Fehlerliste verfolgt und hierdurch den Gesamttestverlauf planen und Nachtests steuern kann. Dieser Prozess der Fehlerkommunikation und Rückmeldung der Fehlerbearbeitung ist in der folgenden Abbildung dargestellt.

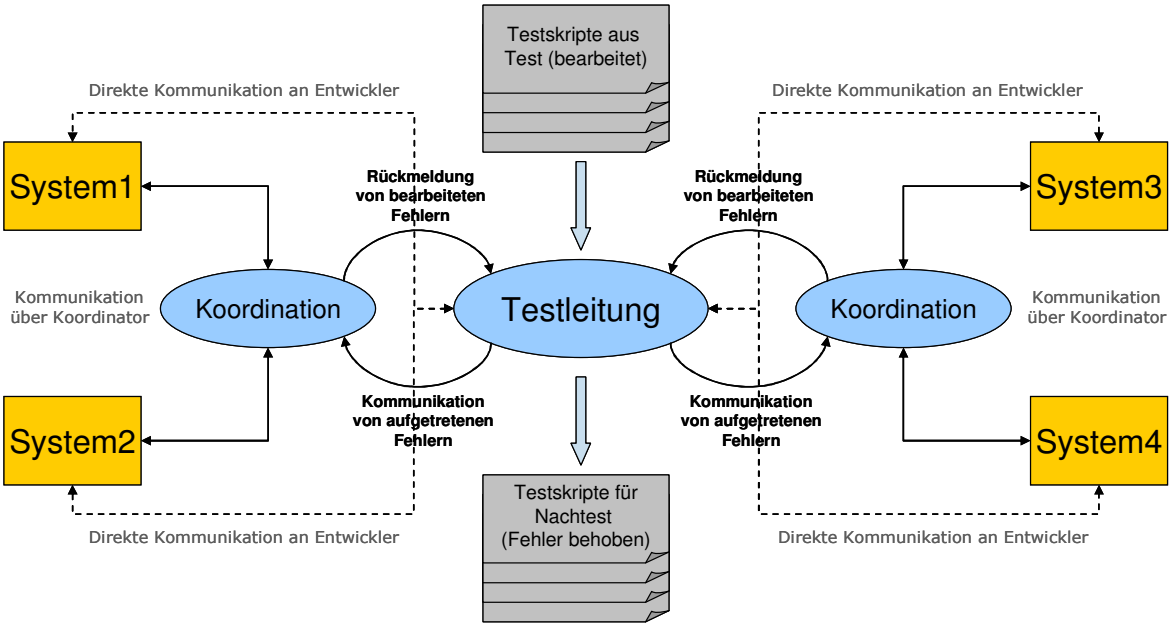


Abbildung 10: Fehlerkommunikation und Fehlerbearbeitung

4.3 Statustracking und -reporting

Ziel des Statustracking ist es, zu jeder Zeit einen Überblick über den Testfortschritt einerseits sowie über die Gesamtheit der aufgetretenen Fehler und ihre Bearbeitungsstati andererseits zu haben und hierüber gegenüber den Auftraggebern auskunftsfähig zu sein. Für die Testfortschrittsverfolgung wird ein spezielles Statustracking Tool auf Basis von Microsoft Excel® verwendet, welches unter [4] als Download verfügbar ist. Dieses Tool ermöglicht auf Testskriptebene die Ergebnisse der aktuellen Ausführung des Testskripts sowie eine Historie zurückliegender Testzyklen darzustellen. Die wichtigsten Informationen können auf dieser Grundlage in Form von Diagrammen visualisiert werden:

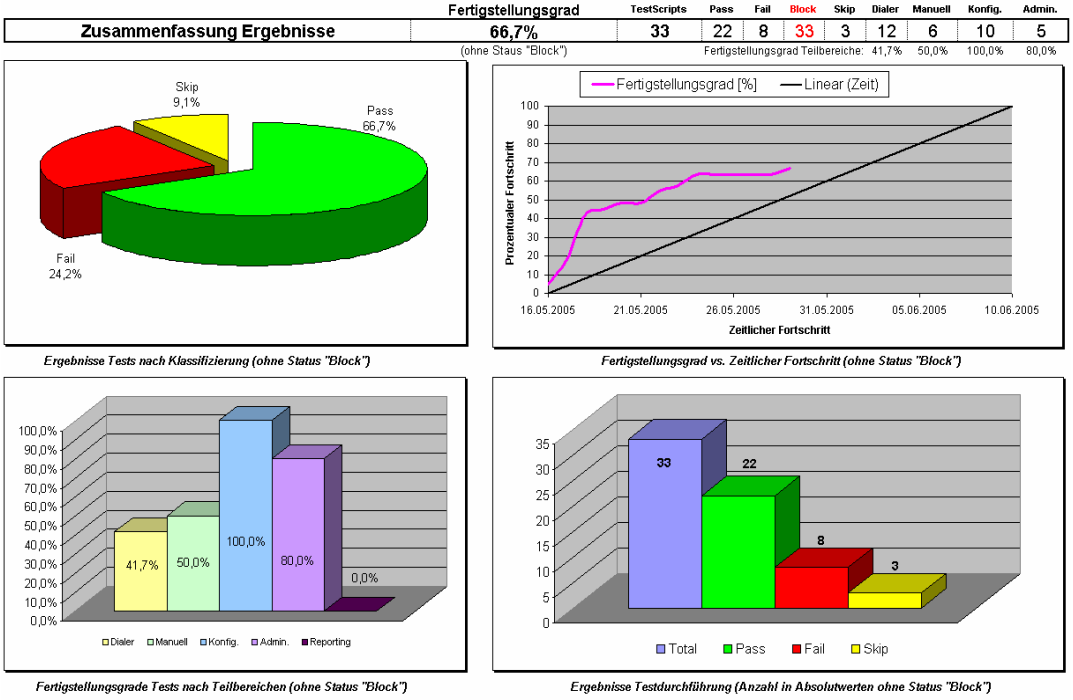


Abbildung 11: Beispiel für ein Statustracking (Testfortschritt)

Ein wesentlicher Bestandteil des Statusreportings ist vor allem der Testfortschritt im zeitlichen Verlauf, gemessen an der Anzahl erfolgreich bearbeiteter Testskripte, sowie der Testskripte in den verschiedenen Bearbeitungsstati (*pass, fail, skip/block* [5]), welche anhand des Statustracking Tools verfolgt und im Rahmen des Reportings berichtet werden können. Im hier beschriebenen Beispiel besteht der Kern des Statusreportings aus einer relativen und absoluten Darstellung des Fertigstellungsgrades (entspricht dem Anteil erfolgreich abgeschlossener Testskripte) im Vergleich zur Gesamtheit der auszuführenden Testskripte, der Entwicklung des Fertigstellungsgrades im zeitlichen Verlauf sowie einer Übersicht der Fertigstellungsgrade nach einzelnen Testabschnitten.

Aus der in Abschnitt 4.2.1.2 vorgestellten Fehlergesamtliste können darüber hinaus Fehlerstatistiken in Form einfacher Excel Standarddiagramme extrahiert werden, welche beispielsweise Fehlerhäufigkeiten, Fehlerverteilungen oder Bearbeitungsstati visualisieren können. Diese können ebenfalls in das Statusreporting mit einfließen (vgl. Abbildung 12).

Schließlich ist es empfehlenswert im Rahmen des Statusreportings auf einer Übersichtsseite die aktuellen Aktivitäten der Planung für den Berichtszeitraum gegenüber zu stellen sowie Risiken, mögliche Verzögerungen im Testbetrieb und den hieraus resultierenden Entscheidungsbedarf gesondert herauszustellen.

---

[5] *pass* = Testskript erfolgreich abgeschlossen; *fail* = Testskript nicht erfolgreich abgeschlossen; *skip* = Test noch nicht ausgeführt; *block* = Test nicht / noch nicht möglich (z.B. wenn eine Funktionalität auf der ein Testskript beruht noch nicht verfügbar ist oder nicht umgesetzt wurde)

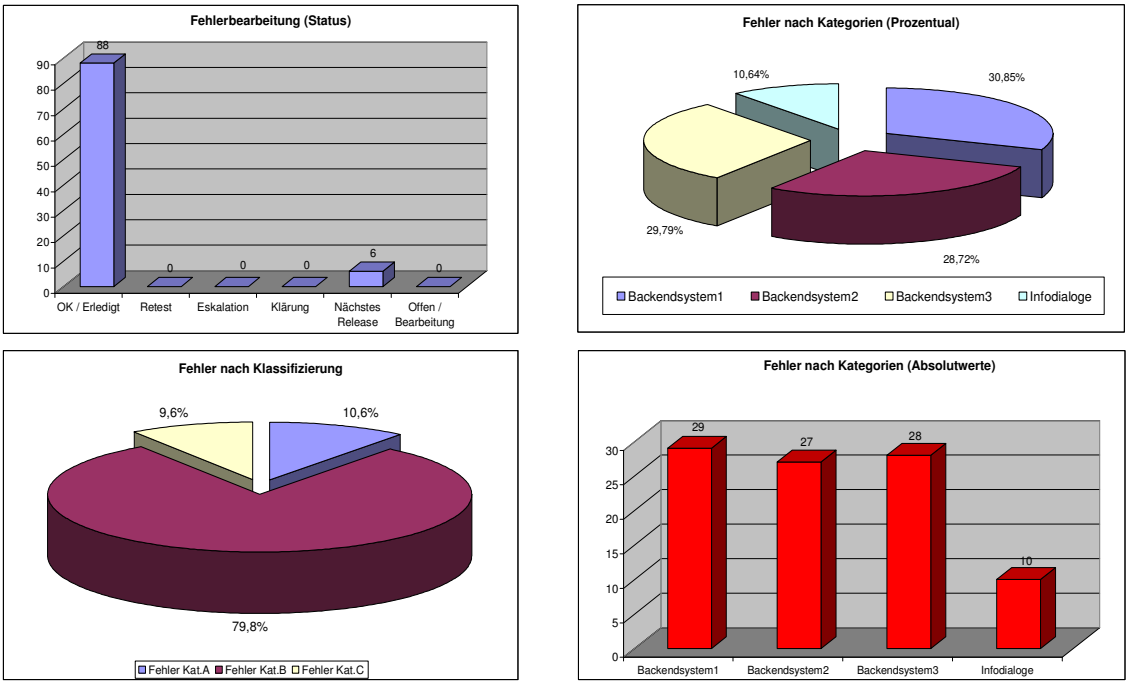


Abbildung 12: Statusreporting – Fehlerstatistiken (exemplarisch)

5 Testmanagement

Der mit der Vorbereitung und Durchführung eines Anwendertests verbundene Aufwand wird oft unterschätzt. Der Testaufwand wächst mit der „Größe“ des Entwicklungsvorhabens, daher wird empfohlen die Verantwortung für den Anwendertest insbesondere im Rahmen größerer Softwareprojekte[6] einer speziell mit den Aufgaben der Testkonzeption, Vorbereitung, Organisation und Ausführung betrauten Testleitung zu übertragen, welche direkt der jeweiligen Projektleitung oder einer vergleichbaren Stelle unterstellt sein und an diese berichten sollte.

5.1 Aufgaben der Testleitung in der Vorbereitungsphase

In der Vorbereitungsphase liegt die Hauptaufgabe der Testleitung in der Konzeptionellen Vorbereitung des Anwendertests sowie in der Bereitstellung der Testorganisation. Details hierzu sind in Abschnitt 3.1ff. beschrieben. Aufgabe der Testleitung ist neben der Testplanung und der Erstellung eines Testkonzepts (vgl. hierzu auch Abschnitt 6.1.1), in welchem der Testumfang, die Vorgehensweise, die Ressourcenplanung usw. dargestellt werden, vor allem die Koordination der unterstützenden Bereiche (z.B. IT Infrastruktur) zur Bereitstellung und Einrichtung der Testumgebung sowie der Bereitstellung der erforderlichen Testdaten, daneben die Erstellung der Testskripte in Zusammenarbeit mit den die spätere Anwendung betreuenden Fachbereichen sowie regelmäßige Berichte an die Projektleitung über den Fortschritt der Testvorbereitung.

---

[6] Genaue Richtlinien können hier nicht vorgelegt werden. Als Anhaltspunkt kann jedoch das Mengengerüst des diesem Bericht zugrunde liegenden Anwendertests dienen. An diesem waren insgesamt 20 Tester in wechselndem Einsatz, drei Entwicklungsteams sowie zwei externe Komponentenlieferanten beteiligt. Der Test erstreckte sich über einen Zeitraum von sieben Monaten (ab Konzeptphase bis zur Produktivsetzung der letzten Systemkomponente bei einer Gesamtprojektlaufzeit von ca. drei Jahren) mit einem Gesamtaufwand für den Test von ca. 440 Personentagen (PT). Hiervon entfielen etwa 180 PT auf die Testressourcen und 150 PT auf die Entwicklungsorganisation (für Bugfixing und Support). Die Testleitung leistete dabei ca. 110 Personentage (Managementaufwand skaliert in der Regel mit der Dauer des Anwendertests). Die Auslagerung der Verantwortung für einen Anwendertest an eine Testleitungsinstanz lohnt sich jedoch auch schon bereits bei deutlich kleineren Projekten.

5.2 Aufgaben der Testleitung während der Testdurchführung

Während der Testdurchführung obliegt der Testleitung die allgemeine Managementfunktion für dieses Teilgebiet des entsprechenden Entwicklungsvorhabens. Hauptaufgabe der Testleitung sind die Steuerung und Überwachung des laufenden Anwendertests sowie das Fehlermanagement (sofern dieses nicht an eine untergeordnete Stelle ausgelagert wurde).
Zur laufenden Testkoordination zählen beispielsweise die regelmäßige Überprüfung und ggf. Anpassung der Testplanung, die Ausgabe zu bearbeitender Testskripte an die Tester, die Entgegennahme, Prüfung, Auswertung und Ablage bearbeiteter Testskripte, laufende Fehlererfassung und Kommunikation an die Entwicklungsorganisation (vgl. Abschnitt 4.2.1.2), sowie Klärung und Lösung von Problemen mit der technischen Infrastruktur (Testumgebung) und sonstiger Probleme aus dem laufenden Testbetrieb mit den zuständigen Stellen. Daneben erfolgen regelmäßige Statusberichte an die Projektleitung sowie regelmäßige Abstimmungen mit den verschiedenen Beteiligten an einem Anwendertest (z.B. Fehlerstatusmeetings mit der Entwicklungsorganisation, Klärung offener fachlicher Fragen mit Fachbereichen, etc.). Nach Beendigung des täglichen Testbetriebs werden in der Regel noch Fehlerlisten bereinigt (z.B. Entfernung gefixter oder redundanter Bugs), sowie die für einen Nachtest in Frage kommenden bzw. für den regulären Test vorgesehenen Testskripte für den Folgetag eingeplant und Testern zugeteilt.
Darüber hinaus erfüllt die Testleitung auch eine Schnittstellenfunktion zur Projektleitung bzw. zu einer vergleichbaren übergeordneten Stelle. Neben regelmäßigen Status- und Fortschrittsberichten werden beispielsweise auch bestehende Probleme, z.B. ein Leerlaufen der „Pipeline“ für Nachtests, an diese zur Eskalation berichtet und spezifische Vorgaben der Auftraggeber, bzw. der Projektleitung an die Testorganisation bzw. die beteiligten Bereiche weitergeleitet und umgesetzt.

6 Testdokumentation

6.1 Bestandteile der Testdokumentation

Die Testdokumentation dient in erster Linie zur Nachvollziehbarkeit der Vollständigkeit sowie zur methodischen Absicherung der Vorgehensweise in einem Anwendertest. Sie sollte sich in Bezug auf Umfang und Inhalte an den Anforderungen der jeweiligen Testsituation orientieren. Hierbei ist zwar darauf zu achten, dass es vor dem Hintergrund der Wirtschaftlichkeit und Effizienz zu keiner Überadministration und keiner unnötigen „Bürokratisierung“ des Tests kommt, dennoch sollten gewisse Standards für die Dokumentation der Konzeption, der Planung, der Testinhalte sowie für die Ergebnisdokumente gelten. Diese Standards für die Testdokumentation werden im Folgenden aus Vorgaben des Industriestandard IEEE/ANSI 829 abgeleitet. Dieser definiert die nachfolgend genannten Arten von Testdokumenten und Berichten. Diese Dokumente werden, sofern sie nicht bereits zuvor Gegenstand der Betrachtung waren, nachfolgend detailliert beschrieben.

Die Testdokumente gliedern sich in:
- Testkonzept
- Testspezifikation
- Testskripte und Testfälle [vgl. Abschnitt 3.1.1]

Die Berichtsdokumente gliedern sich in:
- Fehlermeldungen [vgl. Abschnitt 4.2.1]
- Testprotokolle
- Abschlussbericht

Bei der Erstellung der Testdokumente sollte generell auf ihre Erweiterbarkeit und Wiederverwendbarkeit in ähnlich gelagerten Testsituationen geachtet werden (z.B. in einem Nachtest der Systemfunktionen bei Migrationen oder Modifikationen der Backendsysteme). Bei den hier beschriebenen Vorlagen (z.B. Testskripte) und den folgenden Dokumentationsvorgaben ist dies der Fall, da diese weitestgehend unabhängig von technischen oder organisatorischen Rahmenbedingungen die jeweiligen Sachverhalte beschreiben. Testskripte können direkt wieder verwendet werden, während z.B. das Testkonzept oder die Planungs- bzw. Berichtsdokumente so gehalten sind, dass sie beliebig erweitert und auf neue Testsituationen des entsprechenden Systems angepasst werden können.

6.1.1 Das Testkonzept

Um den Nutzen eines Tests zu maximieren ist eine sorgfältige Planung und Vorbereitung erforderlich, welche im Rahmen des Testkonzepts dokumentiert wird. Das Testkonzept kann nach IEEE/ANSI 829 folgende Komponenten enthalten, welche je nach spezifischer Anforderung der Testsituation übernommen werden sollten (vgl. [3]):

- *Testkonzept ID*: eindeutige Bezeichnung des Dokuments, um es in anderen Dokumenten referenzieren zu können.
- *Einführung*: kurze Zusammenfassung der zu testenden Hard- und Software und der zu testenden Funktionen. Falls es mehrere Testpläne gibt, die hierarchisch geschachtelt sind, dann muss jeder untergeordnete Testplan seinen übergeordneten Testplan referenzieren. Falls die folgenden Dokumente vorhanden sind, sollten sie aufgeführt bzw. referenziert werden:
  - Spezifikationsdokumente
  - Designdokumente
  - Anwenderhandbücher
  - Betriebshandbücher
  - Installationsunterlagen.
- *Zu testende Komponenten*: hier sind sämtliche zu testenden Objekte einschließlich der Versionsnummer bzw. Revisionsnummer aufzuführen. Objekte, die ausdrücklich nicht getestet werden sollen, sollten ebenfalls aufgeführt werden.
- *Zu testende Funktionen*: Benennung sämtlicher zu testenden Funktionen und Funktionskombinationen. Für jede zu testende Funktion bzw. Funktionskombination ist das dazugehörige Testskript (Referenz/ID) anzugeben. Hiervon abgegrenzt werden Funktionen, die nicht getestet werden.
- *Vorgehensweise*: Beschreibung des allgemeinen Testvorgehens. Die Aktivitäten und zum Einsatz kommenden Techniken und Werkzeuge sind zu spezifizieren. Das Vorgehen sollte so detailliert beschrieben werden, dass es möglich ist, auf dieser Grundlage eine Aufwandsschätzung abzugeben. Ende-/Abbruchkriterien sind zu definieren. Weiterhin sind alle bedeutenden und bekannten Einschränkungen des Testvorgangs aufzuführen, wie Verfügbarkeit der Testobjekte, Ressourcen und Termine.
- *Kriterien für erfolgreiche bzw. fehlgeschlagene Testläufe* sind zu beschreiben (pass/fail Kriterien).
- *Kriterien für Testunterbrechungen und -wiederaufnahme*: hier sind die Kriterien zu definieren, die eine Unterbrechung aller oder eines Teils der Aktivitäten des Testplans bedingen. Außerdem ist zu definieren, welche Aktivitäten bei Wiederaufnahme des Testbetriebs wiederholt werden müssen.
- *Produkte*: hierunter werden die im Testverlauf zu erstellenden Dokumente aufgeführt, beispielsweise Testkonzept, Testskripte/-fälle, Testlogs, Testprotokolle (Ergebnisse), Abschlussbericht.
- *Testtätigkeiten*: hier sind alle Tätigkeiten zu beschreiben, die zur Vorbereitung und zur Durchführung der Tests nötig sind. Alle Abhängigkeiten zwischen den Tätigkeiten und alle erforderlichen Spezialkenntnisse sind zu identifizieren.
- *Testumgebung*: Spezifikation der notwendigen sowie der wünschenswerten Eigenschaften der Testumgebung. Diese sollte die wesentlichen physischen Charakteristiken der Hardware und Software sowie weitere Betriebsmittel zur Unterstützung der Tests beinhalten. Wenn spezielle Tools benötigt werden, sind diese aufzulisten, ebenso weitere Anliegen, wie z.B. Büroraum oder Fachliteratur. Für alle Materialien, die im Moment noch nicht zur Verfügung stehen, ist eine Quelle anzugeben.
- *Verantwortlichkeiten*: hier sind die verantwortlichen Gruppen für Management, Design, Vorbereitung, Ausführung, Beglaubigung, Überprüfung und Problemlösung zu benennen.
- *Ressourcen*: Personalbedarf für die Testaufgabe nach erforderlichen Skills geordnet.
- *Zeitplan*: im Zeitplan müssen sowohl die Test Meilensteine aus der Software Projektplanung als auch die Lieferungen der zu testenden Objekte festgehalten werden. Auch zusätzliche Meilensteine können aufgeführt werden. Für jede Testaufgabe ist der Zeitbedarf zu schätzen sowie ein Zeitplan anzugeben. Für jede Ressource (Geräte, Tools, Personal) sollte man den Betriebs- bzw. Bedarfszeitrahmen definieren.
- *Risiken und Risikomanagement*: es sind alle mit hohem Risiko behafteten Annahmen des Testplans aufzuführen. Für die Annahmen sind geeignete eine Notfallmaßnahmen zu definieren (z.B. verspätete Lieferung von zu testenden Objekten kann zu Verzögerungen im Testbetrieb führen und somit den geplanten Produktivsetzungstermin gefährden – Notfallmaßnahme: Längere Arbeitszeiten im Anwendertestbetrieb).

- *Genehmigungen*: Namen und Titel aller Personen, die diesen Testplan genehmigen müssen. Zu Dokumentationszwecken sollte eine gedruckte Kopie des Testplans unterzeichnet und archiviert werden.

6.1.2 Testspezifikation

Die Testspezifikation kann entweder ein separates Dokument sein oder ein Bestandteil des Testkonzepts. Die Testspezifikation enthält im Wesentlichen eine detaillierte Auflistung aller zu testenden Funktionen und Funktionskombinationen mit Referenzen zu den zugrunde liegenden Spezifikationen im Fach-/Grobkonzept. Darüber hinaus wird hier das Testverfahren detailliert dargestellt, spezielle Test- und Analysetechniken sind, sofern sie zum Einsatz kommen, zu beschreiben. Schließlich sind hier noch einmal alle Testskripte/Testfälle aufzulisten, gemeinsam mit den definierten pass/fail Kriterien.

6.1.3 Testprotokolle und Abschlussbericht

Testprotokolle enthalten Angaben über alle ausgeführten Testfälle und deren Ergebnisse. Sie dienen nicht nur dazu, alle aufgetretenen Fehler für die Entwicklungsorganisation zu dokumentieren, sondern auch dazu, die korrekte Arbeit des Testteams zu belegen. Treten z.B. nach der Produktivsetzung noch Fehler auf, lässt sich anhand einer vollständigen Testprotokollierung nachweisen, ob für die fragliche Situation ein Testfall existiert, ob er ausgeführt wurde und was das Ergebnis war.
Die Testprotokollierung erfolgt durch das in Abschnitt 4.3 beschriebene Statustracking im Zusammenhang mit der durch die verantwortlichen Bereiche abgenommenen und freigegebenen Gesamttestplanung, welche sämtliche auszuführenden Testskripte enthält (vgl. Abbildung 5). Schließlich sind nach Beendigung des Tests sämtliche ausgeführten und von den Testern nach Ausführung abgezeichneten Testskripte zu Dokumentationszwecken zusammen mit der übrigen Testdokumentation zu archivieren.

Um den Auftraggebern nach Abschluss des Tests einen fundierten und detaillierten Überblick über die Testergebnisse zu ermöglichen wird das Ergebnis der Testaktivitäten in einem Abschlussbericht zusammengefasst, welcher in erster Linie eine Abschätzung der Gründlichkeit und Güte der Tests erlauben sollte. Der Abschlussbericht enthält nach IEEE/ANSI 829 folgende Komponenten, welche je nach spezifischer Anforderung der Testsituation übernommen werden sollten (vgl. [3]):

- *Abschlussbericht ID*: eindeutige Bezeichnung, um den Abschlussbericht in anderen Dokumenten identifizieren zu können.
- *Zusammenfassung*: Zusammenfassung des Testergebnisses. Identifikation der getesteten Objekte mit Hinweis auf Versionen und Revisionen. Beschreibung des Testumfangs der Vorgehensweise und des Testzeitraums, ggf. mit Verweisen auf zugrunde liegende Testpläne, Testspezifikationen, Testskripte, Protokolle, Statusberichte und Fehlermeldungen.
- *Abweichungen*: Bericht über Abweichungen der Software von ihren vorgesehenen Spezifikationen. Beschreibung auch von Abweichungen vom Testplan oder der Testspezifikation sowie der Gründe hierfür.
- *Testumfang*: Darstellung der Vollständigkeit des Testumfangs. Darstellung von Funktionalitäten, welche nicht oder nicht ausreichend getestet wurden, und Benennung der Gründe hierfür.
- *Testergebnisse*: Zusammenfassung der Testergebnisse. Darstellung von Fehlerhäufigkeiten und Fehlerverteilungen in aggregierter Form. Beschreibung des Fehlermanagementprozesses und der Ergebnisse der Fehlerbearbeitung. Darstellung der Ergebnisse der einzelnen Testzyklen und des Testfortschritts im Zeitverlauf. Explizite Benennung der nach Testende verbleibenden Fehler und offenen Punkte.
- *Bewertung*: Allgemeine Einschätzung des getesteten Systems in Bezug auf die gewünschten Qualitätskriterien und eventuelle Einschränkungen basierend auf den Testergebnissen. Verzögerungen oder Erschwernisse des Testbetriebs können hier dargestellt werden. Die Bewertung sollte darüber hinaus eine Einschätzung der verbleibenden Fehlerrisiken enthalten.
- *Aktivitäten*: Zusammenfassung der Hauptaktivitäten im Testbetrieb sowie der wichtigen Ereignisse im Testverlauf, ggf. ergänzt um Berichte über den Ressourceneinsatz und -verbrauch.
- *Genehmigungen*: angesichts der Bedeutung des Abschlussberichts im Rahmen einer formalen Abnahme bzw. Freigabe des Tests sollte dieser ausdruckt, von den hier bezeichneten Personen unterschrieben und anschließend archiviert werden.

7 Zusammenfassung / Fazit

Keine Software und kein auf Software basierenden System funktioniert von vornherein genau so, wie es auf Basis seiner Spezifikation entwickelt wurde. Im Laufe der Programmierung geschehen immer Fehler, und je komplexer ein System ist, desto größer und gravierender sind die zu erwartenden Fehler. Erfolgreiche Anwendertests sind daher eine wesentliche Grundlage für den stabilen und weitestgehend störungsfreien Produktivbetrieb eines Softwaresystems. Nachfolgend werden noch einmal die wesentlichen Erfolgsfaktoren für effiziente und effektive Anwendertests zusammengefasst:

1. *Trennung der Testphasen und definierte Übergange*:
   Ein erfolgreicher Anwendertest setzt einen ebenso gründlich und vollständig ausgeführten Integrationstests voraus und dieser wiederum entsprechende Komponententests. Fehler aus der Integrationsphase, welche in dieser nicht erkannt und behoben wurden können einen Anwendertest erheblich erschweren. Von daher ist die gleiche methodische Gründlichkeit in jeder Testphase gleichermaßen anzusetzen und bei jedem Übergang in eine neue Testphase wird ein Expertenreview mit formaler Freigabe der Ergebnisse empfohlen.
2. *Testskripte*:
   Die Testskripte sind der Schlüssel zur Ausführung des Anwendertests. Hierbei ist vor allem zu beachten, dass sämtliche im Fachkonzept spezifizierten Funktionalitäten durch die Testskripte abgedeckt werden, dass die Testskripte klar und deutlich strukturiert und formuliert sind sowie einfach in der Handhabung und Verwaltung.
3. *Testplanung, -organisation und Beteiligung der Fachbereiche*:
   Der Aufwand eines Anwendertests ist nicht zu unterschätzen. Neben dem Aufwand für die Ausführung der Testskripte sollte auch Zeit für die Fehlerbehebung, erforderliche Abstimmungen, eventuelle Verzögerungen etc. eingeplant werden. Auch ist wichtig, dass die von einer Systemeinführung betroffenen (Fach-)bereiche in allen Phasen des Anwendertests (Planung, Entwicklung der Testskripte, Testausführung) aktiv beteiligt werden.
4. *Fehlermanagement*:
   Ein effizientes Fehlermanagement (Erfassung, Priorisierung, Kommunikation, Rückmeldung und Nachtest) ist entscheidend für die schnelle Bearbeitung von im Testbetrieb auftretenden Fehlern. Hier sind geeignete Tools zur schnellen und effizienten Erfassung und Kategorisierung sowie für die effektive Fehlerkommunikation mit der Entwicklungsorganisation zu wählen.
5. *Testmanagement:*
   Dem Testmanagement obliegt die gesamte Verantwortung für die Vorbereitung und Durchführung des Anwendertests. Es wird empfohlen diese Aufgabe nicht in der Projektleitung oder Linienorganisation anzusiedeln sondern einer der Projektleitung direkt untergeordneten und an diese berichtenden Stelle zu übertragen.
6. *Dokumentation*:
   Eine durchgehende, standardisierte Testdokumentation, welche alle Phasen des Anwendertest, von der Konzeption bis zum Abschlussbericht umfasst, gewährleistet eine methodische Absicherung sowie die spätere Nachweismöglichkeit, dass spezifische Funktionalitäten getestet wurden, gemeinsam mit dem Nachweis des jeweiligen Ergebnisses.

Abschließend sei erwähnt dass der diesem Bericht zugrunde liegende Anwendertest auf der Grundlage der hier dargestellten Inhalte sehr erfolgreich durchgeführt werden konnte. Die anschließende Systemeinführung (Produktivsetzung) verlief weitgehend reibungslos und die Anlaufunterstützung durch die Testorganisation konnte nach nur zwei Wochen eingestellt werden. Seitdem ist das System ohne größere Fehler im Produktivbetrieb im Einsatz.