

Referenzprofil

IT Test Coordinator

Armin Saalman

Dieses Referenzprofil wurde im Rahmen des bmb+f geförderten Projekts „Arbeitsprozess-orientierte Weiterbildung in der IT-Branche“ erarbeitet von:



Fraunhofer ISST

Bildungspartner

Unternehmenspartner

Danksagung

Diese Profilbeschreibung entstand auf Basis mehrerer Praxisprojekte der Firma *imbus*, deren Vorstand Herrn Tilo Linz und dem Projektleiter Thomas Rumi wir herzlich für ihre fachkundige und umfassende Hilfe danken. Fachlich beratend mitgewirkt hat Herr Hodina von *Lutz & Grub*. Ohne ihre Mithilfe hätte dieses Dokument nicht entstehen können.

Inhalt

1	EINFÜHRUNG: REFERENZPROZESSE ALS CURRICULA	4
1.1	EREIGNIS-PROZESS-KETTEN: SYMBOLIK	5
1.2	REFERENZPROZESS UND TEILPROZESSE	7
2	DAS PROFIL: IT TEST COORDINATOR (IT-TESTKOORDINATOR/IN)	9
2.1	TÄTIGKEITSBESCHREIBUNG	9
2.2	PROFILTYPISCHE ARBEITSPROZESSE	10
2.3	PROFILPRÄGENDE KOMPETENZFELDER	11
2.4	QUALIFIKATIONSERFORDERNISSE	12
2.5	EINORDNUNG INS SYSTEM UND KARRIEREPFADE	12
3	REFERENZPROZESS	13
3.1	KERNTÄTIGKEITEN DES IT TEST COORDINATOR	13
3.1.1	Referenzprozess IT Test Coordinator	15
3.1.2	Die Beispielprojekte	17
3.1.2.1	Beispielprojekt für den Modultest nach dem Black-Box-Verfahren	17
3.1.2.2	Beispielprojekt für den Integrationstest	17
3.1.2.3	Beispielprojekt für den Systemtest – Schwerpunkt Installation und Lizenzierung	18
3.1.2.4	Überblick über die Beispielprojekte	18
3.1.3	Prozesskompass IT Test Coordinator	20
3.1.4	Teilprozesse des IT Test Coordinator	21
3.1.4.1	Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept	22
3.1.4.2	Instanzieren des Modultests (als White Box Test)	26
3.1.4.3	Spezifizieren des Modultests (als White Box Test)	28
3.1.4.4	Implementieren des Modultests (als White Box Test)	30
3.1.4.5	Instanzieren des Modultests (als Black Box Test)	32
3.1.4.6	Spezifizieren des Modultests (als Black Box Test)	34
3.1.4.7	Implementieren des Modultests (als Black Box Test)	36
3.1.4.8	Durchführen des Modultests (als Black Box Test)	38
3.1.4.9	Instanzieren des Integrationstests	42
3.1.4.10	Spezifizieren des Integrationstests	44
3.1.4.11	Implementieren des Integrationstests	46
3.1.4.12	Durchführen des Integrationstests	48
3.1.4.13	Instanzieren des Systemtests	51
3.1.4.14	Spezifizieren des Systemtests	54
3.1.4.15	Implementieren des Systemtests	57
3.1.4.16	Mitwirken beim Systemtest	59
3.1.4.17	Kategorisieren der Fehler/Erstellen des Testberichts	63
4	GLOSSAR	65

1 Einführung: Referenzprozesse als Curricula

Das Referenzprojekt des IT Test Coordinator (IT-Testkoordinator/in) verdeutlicht paradigmatisch die diesem Tätigkeitsfeld zugrunde liegenden Arbeitsprozesse, die mit ihnen verbundenen Ansprüche sowie die daraus resultierenden Anforderungen an Inhalt und Durchführung einer qualitativ hochwertigen Weiterbildung.

Das Referenzprojekt erfüllt mehrere Funktionen:

Aus der Praxis für die Praxis

Als Abstraktion tatsächlich stattgefundener Projekte und Prozesse bieten die Referenzprozesse eine realistische und leicht nachvollziehbare Abbildung dessen, was die Tätigkeiten eines IT Test Coordinator sind.

Prozessorientierung als innovatives „Curriculum“

Als vollständige Darstellung aller wichtigen Arbeitsprozesse sowie der dazugehörigen Qualifikationen, Tätigkeiten und Werkzeuge bieten die Referenzprozesse die Grundlage für die Weiterbildung zum IT Test Coordinator. Alle diese Prozesse müssen – entsprechend den Vorgaben – einmal oder mehrfach durchlaufen werden und ermöglichen dadurch den Weiterzubildenden den arbeitsplatznahen, integrativen Erwerb von relevanten Kompetenzen. Durch den Verbleib im Arbeitsprozess wird nicht nur für die Weiterzubildenden eine hohe Motivation (Arbeit an echten Projekten/Aufgaben) und Nachhaltigkeit erreicht, sondern auch – aus Sicht des Unternehmens – die Kontinuität und Qualität der laufenden Arbeiten gesichert (keine Ausfallzeit durch Seminartage, kein mühsamer Transfer).

Qualitätsstandard für die Weiterbildung

Als Referenz bieten insbesondere die Teilprozesse und die mit ihnen verbundenen Tätigkeits- und Qualifikationsziele einen Qualitätsmaßstab für die arbeitsprozessorientierte Weiterbildung und die resultierenden Abschlüsse. Vollständige Transparenz und klare Zielvorgaben ermöglichen die qualitativ hochwertige Absicherung auch komplexer Kompetenzen sowie den systematischen Erwerb des notwendigen Erfahrungswissens.

Transferprozesse

Die Generalisierung des Referenzprojekts aus der Praxis und seine didaktische Anreicherung ermöglichen eine leichte Auswahl angemessener Transferprozesse, deren Bearbeitung die Grundlage der Weiterbildung ist. Transferprozesse sind reale Prozesse, die Referenzprojekte in einer lernförderlichen Umgebung abbilden. Abgeschlossene Transferprozesse auf Basis der hier dargestellten Anforderungen und Qualitätsmaßstäbe sind nicht nur Qualifikationsnachweis des Einzelnen, sondern bilden auch die Basis eines angemesseneren und zielgerichteteren Umgangs mit Geschäfts- und Arbeitsprozessen im Unternehmen.

1.1 Ereignis-Prozess-Ketten: Symbolik

Die Darstellung der Referenzprozesse in Form von Ereignis-Prozess-Ketten¹ ermöglicht einen schnellen Überblick. Die Vollständigkeit kann leicht überprüft werden, Anpassungen und Modifikationen in Hinblick auf das eigene Unternehmen sind problemlos möglich und Anknüpfungspunkte an andere Prozesse, aber auch zu weiterführenden Informationen ergeben sich automatisch.

Die bei der Darstellung der Referenz- und Teilprozesse verwendete Modellierungssprache stellt eine Anpassung und Weiterentwicklung der klassischen EPK-Modellierung dar:

- Referenz- wie Teilprozesse sind aus der Sicht des jeweiligen Spezialisten, also als Arbeitsprozesse einer Person dargestellt.
- Referenz- wie Teilprozesse stellen in der Regel keinen Geschäftsprozess dar.

Die EPK-Symbole werden hier wie folgt verwendet:

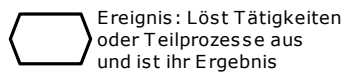
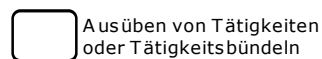
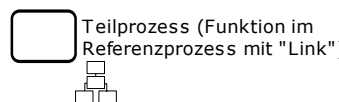


Abbildung A: Grundlegende Symbole der Referenz- und Teilprozessmodelle.

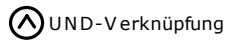
Die wichtigsten Symbole sind:

- die Tätigkeiten bzw. Tätigkeitsbündel oder Teilprozesse, die mit dem Funktionssymbol dargestellt werden
- die Ereignisse, die Tätigkeiten bzw. Teilprozesse auslösen und Ergebnisse von Teilprozessen sind

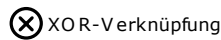
Grundsätzlich gilt: Auf ein Ereignis folgt immer ein Teilprozess bzw. eine Tätigkeit.

Ergebnisse von Tätigkeiten sind sehr oft Dokumente; diese werden dann zusätzlich durch das Dokumentsymbol dargestellt.

¹ Vgl. A.-W. Scheer, *Wirtschaftsinformatik*, Springer 1998.



UND-Verknüpfung



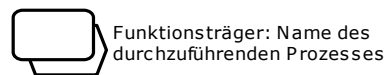
XOR-Verknüpfung



ODER-Verknüpfung

Abbildung B: Konnektoren.

Wenn Alternativmöglichkeiten bestehen, werden Ereignisse und Teilprozesse/Tätigkeiten über Konnektoren (AND, OR, XOR) verbunden. Dabei steht AND für ein verbindendes „Und“, OR für ein „Oder“, das alle Möglichkeiten offen lässt, und XOR für ein „ausschließendes Oder“, welches nur einen der angegebenen Pfade ermöglicht.



Funktionsträger: Name des durchzuführenden Prozesses

Abbildung C: Schnittstelle.

Da die Prozesse aus der Sicht des jeweiligen Spezialisten formuliert werden, sind Schnittstellen zu Prozessen anderer Spezialisten oder zu Entscheidungsprozessen auf höherer Ebene notwendig. Dazu wird das Schnittstellensymbol verwendet. Es steht für Prozesse, die der Spezialist nicht selber durchführt, auf deren Durchführung er aber angewiesen ist. Parallel zu jeder Schnittstelle wird die Tätigkeit dargestellt, die der Spezialist selbst in diesem Zusammenhang ausübt, wie „Beraten bei ...“, „Unterstützen bei ...“ oder „Informieren des ...“.

Alle Prozesse werden durch die Verwendung dieser Symbole klar und einfach strukturiert dargestellt und sind offen für die Übertragung in konkrete Transferprozesse.

1.2 Referenzprozess und Teilprozesse

Der hier vorgestellte Referenzprozess und seine Teilprozesse stellen das Curriculum des Spezialistenprofils „IT Test Coordinator“ dar.

Der Referenzprozess erhebt nicht den Anspruch eines Vorgehensmodells, sondern bildet beispielhaft den möglichen Arbeitsprozess und Verlauf eines Projekts auf Spezialistenebene ab.

Er bildet die Grundlage für Weiterbildungen und damit einen Qualitäts-, Niveau- und Komplexitätsmaßstab. Die zugehörigen Teilprozesse sind hier beispielhaft modelliert und stellen eine Möglichkeit der Durchführung dar. Einzelheiten zu den unverzichtbaren Prozessen und Kompetenzfeldern sind hier im Referenzprojekt festgelegt. Die Reihenfolge und die Inhalte der Teilprozesse sind abhängig vom jeweils auszuwählenden Transferprojekt und werden in diesem Zusammenhang festgelegt.

Die Darstellung der Prozesse erfolgt systematisch:

Jeder Prozess wird mithilfe von Ereignis-Prozess-Ketten dargestellt. Einem auslösenden Ereignis folgt eine Funktion, die wiederum ein oder mehrere Ereignisse als Ergebnis hat. Ereignisse und Funktionen können mit AND, OR oder XOR, den Konnektoren, verbunden sein.

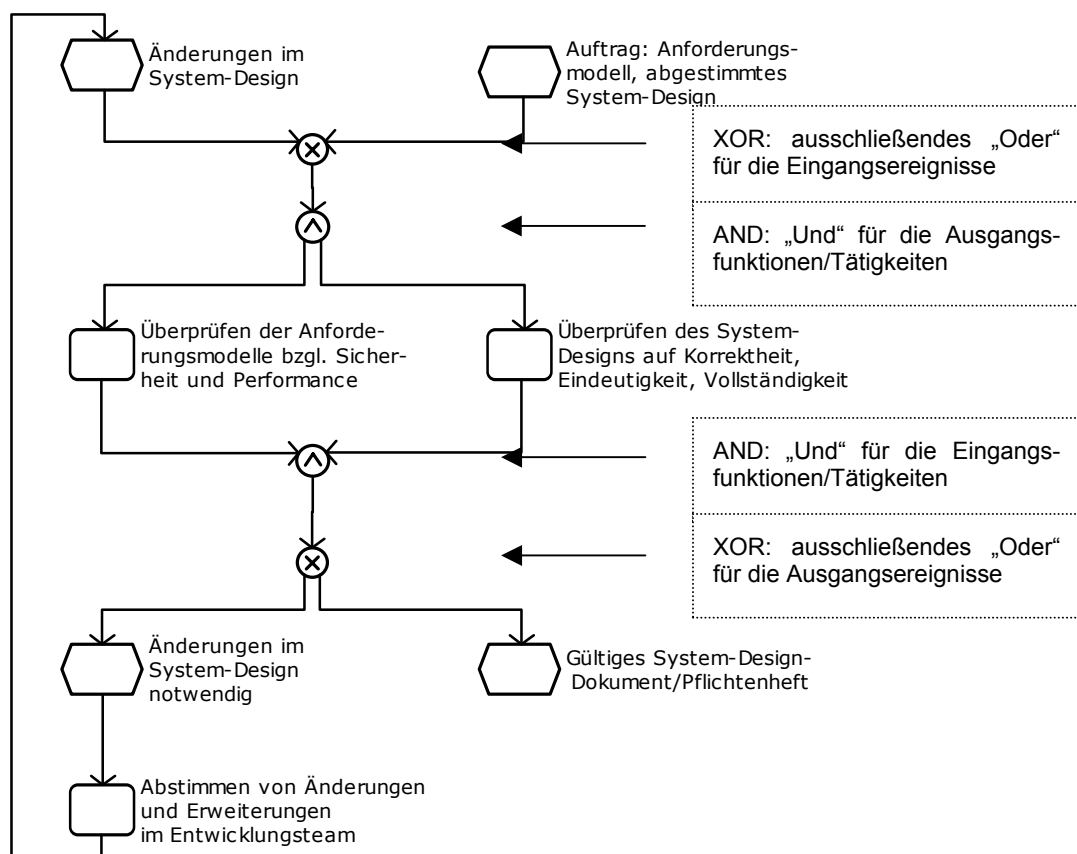


Abbildung D: Beispielprozess (Teilprozess "Überprüfen des Systemdesigns") mit unterschiedlicher Verwendung von Konnektoren.

Die Verbindung von Referenzprozess und Teilprozessen erfolgt über die Funktionen des Referenzprozesses:

Jede Funktion im Referenzprozess steht für einen Teilprozess.

Ereignisse, die dem jeweiligen Teilprozess direkt vor- oder nachgeordnet sind, sind Anfangs- und Endereignisse der jeweiligen Teilprozesse. Damit stellen die Teilprozesse die Funktio-

nen des Referenzprozesses ausführlich dar, und ein Hin- und Herbewegen zwischen Referenz- und Teilprozessen ist jederzeit problemlos möglich.

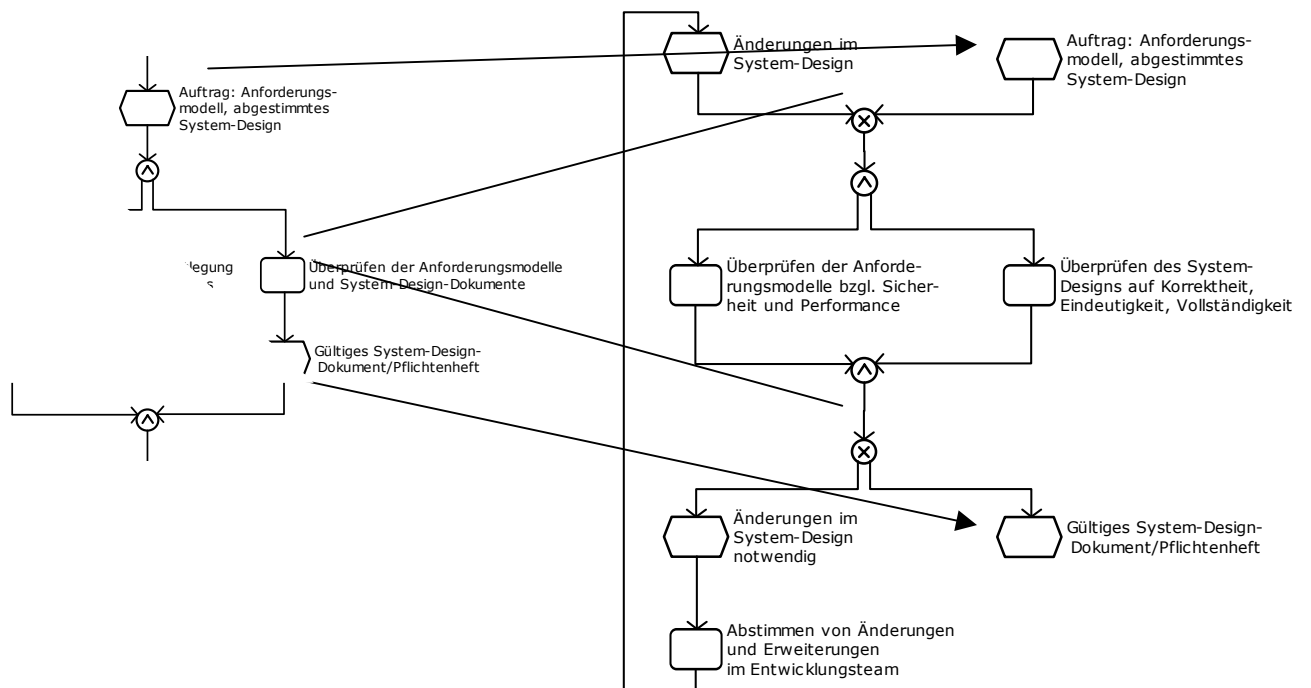


Abbildung E: Ausschnitt aus dem Referenzprozess des Database Developer (links) und Teilprozess des Database Developer "Überprüfen des Systemdesigns" (rechts).

Die Teilprozesse stellen so die wesentlichen Teile eines Projekts dar und lassen sich entsprechend auf Transferprojekte übertragen. Den Teilprozessen sind die jeweils wesentlichen Tätigkeiten und Kompetenzfelder zugeordnet.

2 Das Profil: IT Test Coordinator (IT-Testkoordinator/in)

IT Test Coordinator² konzipieren die Tests, die den Software- und Hardware-Entwicklungsprozess begleiten, auf den Stufen Modul-, Integrations- und Systemtest und führen diese Tests durch. Sie stellen Testumgebungen bereit und sind für die Tests auf allen Teststufen verantwortlich. Dabei führen sie das Testprotokoll und erzeugen im Fehlerfall ein Fehlerprotokoll, das an die Entwickler weitergeleitet wird. Zum Abschluss eines Release-Zyklus mit mehreren Testzyklen liegt ein Release vor, über das der IT Test Coordinator einen Testbericht auf der Basis des Testprotokolls erstellt.

2.1 Tätigkeitsbeschreibung

IT Test Coordinator begleiten und unterstützen den Soft- und Hardware-Entwicklungsprozess in enger Zusammenarbeit mit Kunden und den Spezialisten aus den Bereichen Systemanalyse, Systementwicklung und Produktion durch angemessene und aussagekräftige Tests. Grundlegend ist dafür ein hohes Verständnis des Produkts oder der Systemlösung und der gestellten Anforderungen und Vorgaben.

Zur Testentwicklung gehören der Entwurf und die Definition von Teststrategien, Testdaten, Testfällen und Testszenarien; die Planung und das Design von Testumgebungen; die Implementierung und/oder Instanziierung von Tests; die Erstellung von automatisierten Testsuites für Black- und White Box Tests; die Durchführung manueller oder automatischer Tests auf den Ebenen Modul-, Integrations- und Systemtests; die Testprotokollierung sowie Aufbau, Parametrierung und Wartung von Testumgebungen.

Des Weiteren arbeiten Test Coordinator konstruktiv mit den Spezialisten aus den Bereichen Technik (insbesondere dem Component Technician und Industrial Systems Technician) und Software-Entwicklung (insbesondere dem Projektleiter bei der Erstellung des Test- und Entwicklungsplans) zusammen und erarbeiten mögliche Prozessverbesserungen (Testdurchführung, Hard-/Software-Entwicklung) für Entscheidungen auf der Professional-Ebene.

Neben umfangreichen Erfahrungen in der Entwicklung von Hardware oder Software verfügen IT Test Coordinator über umfangreiche Fähigkeiten und Fertigkeiten im Bereich des Testmanagements, der Teststrategien, Testumgebungen, Testdurchführung und Testauswertungen.

² Kapitel 2: „Das Profil: IT Test Coordinator (IT-Testkoordinator/in)“ gibt den offiziellen Text der „Vereinbarung über die Spezialistenprofile im Rahmen des Verfahrens zur Ordnung der IT-Weiterbildung“ vom 25.05.2002 (Bundesanzeiger 105, ausgegeben am 12.06.2002) wieder.

2.2 Profiltypische Arbeitsprozesse

Die im Folgenden beschriebenen Teilprozesse dokumentieren den gesamten profiltypischen Arbeitsprozess des IT Test Coordinator. Die Beherrschung dieses Arbeitsprozesses in Verbindung mit den Kompetenzen in den jeweiligen Kompetenzfeldern und der Berufserfahrung bildet die Grundlage für die berufliche Handlungskompetenz.

1. Beim Mitwirken bei der Erstellung einer Projektdefinition und des Testplans werden zum Beispiel die Schritte Festlegen der Testzyklen, Festlegen der Testmethoden und Festlegen der Testziele durchlaufen.
2. Das Instanzieren des Modultests (White Box Test) beinhaltet die Identifizierung der Testfälle, Festlegen der Testabbruchbedingungen und Festlegen der Testanforderungen.
3. Das Spezifizieren des Modultests (White Box Test) beinhaltet das Festlegen der Testmethode und der Testobjekte, die Überprüfung der Testfälle auf eine Testautomation, das Festlegen der Testdaten für die Testfälle und die Überprüfung der Testfälle gegen die Testanforderungen und Ziele.
4. Das Implementieren des Modultests (White Box Test) beinhaltet die Implementierung der Testtreiber in Zusammenarbeit mit den Software-Entwicklern des Moduls und die Erstellung entsprechender Testskripte.
5. Das Instanzieren des Modultests (Black Box Test) besteht aus den Schritten Identifizierung der Testfälle, Festlegen der Testabbruchbedingungen und Festlegen der Anforderungen an die Testumgebung.
6. Das Spezifizieren des Modultests (Black Box Test) beginnt mit der Festlegung der Testdaten für jeden Testfall des Modultests. Anschließend wird der Testfall auf Testautomation überprüft.
7. Das Implementieren des Modultests (Black Box Test) besteht aus dem Implementieren der Testtreiber für den Modultest, dem Erstellen der Testumgebung und ggf. Erstellen der Testskripte.
8. Das Durchführen des Modultests (Black Box Test) beginnt mit der Auswahl des durchzuführenden Testfalls mit anschließender manueller oder automatisierter Durchführung. Danach folgt im Fehlerfall die Erstellung eines Fehlerprotokolls und zum Abschluss die Überprüfung des Testergebnisses gegen die Testabbruchbedingungen.
9. Das Instanzieren des Integrationstests beinhaltet die Tätigkeiten Festlegen der Testabbruchbedingungen, Festlegung der Anforderungen an die Testumgebung und Erstellen der Testfälle.
10. Das Spezifizieren des Integrationstests enthält die Arbeitsschritte Erstellen der Testdaten und Überprüfen des Testfalls auf Testautomation.
11. Das Implementieren des Integrationstests erfolgt in den Schritten Implementierung der Testtreiber, Erzeugung der Testumgebung für den Integrationstest und Erstellung der Testskripte für die automatischen Testfälle.
12. Das Durchführen der Integrationstests beginnt mit der Auswahl der priorisierten Testfälle; danach erfolgt die Ausführung der Tests mit der Überprüfung der Testergebnisse. Im Fehlerfall wird ein Fehlerprotokoll erstellt.
13. Das Instanzieren des Systemtests erfolgt in den Schritten Festlegung der Anforderungen an die Systemtestumgebung und der Testabbruchkriterien wie zum Beispiel maximale Anzahl schwerer Fehler beim Systemtest.
14. Das Spezifizieren des Systemtests beginnt mit der Entwicklung der Testfälle und der Festlegung der Testdaten für jeden Testfall des Systemtests. Anschließend wird der Testfall überprüft, ob eine Testautomation sinnvoll ist.
15. Das Implementieren des Systemtests erfolgt in den Schritten Implementierung der Testtreiber, Erstellung der Testskripte und Erzeugung der Testumgebung für den Systemtest.

16. Das Mitwirken beim Systemtest besteht aus der manuellen oder automatischen Testdurchführung, dem Erstellen eines Fehlerprotokolls im Fehlerfall und dem Führen des Testprotokolls.
17. Das Kategorisieren der Fehler enthält Tätigkeiten zur Auswertung des Testprotokolls wie zum Beispiel Erstellen einer Fehlerstatistik bis auf Testfallebene oder Erstellung des Testberichts.

2.3 Profilprägende Kompetenzfelder

Die Beherrschung der profiltypischen Arbeitsprozesse setzt Kompetenzen unterschiedlicher Reichweite in den nachstehend aufgeführten beruflichen Kompetenzfeldern³ voraus. Den Kompetenzfeldern sind Wissen und Fähigkeiten sowie typische Methoden und Werkzeuge unterschiedlicher Breite und Tiefe zugeordnet.

Grundlegend zu beherrschende, gemeinsame Kompetenzfelder⁴:

- Unternehmensziele und Kundeninteressen
- Problemanalyse, -lösung
- Kommunikation, Präsentation
- Konflikterkennung, -lösung
- fremdsprachliche Kommunikation (englisch)
- Projektorganisation, -kooperation
- Zeitmanagement, Aufgabenplanung und -priorisierung
- wirtschaftliches Handeln
- Selbstlernen, Lernorganisation
- Innovationspotenziale
- Datenschutz, -sicherheit
- Dokumentation, -standards
- Qualitätssicherung

Fundiert zu beherrschende, gruppenspezifische Kompetenzfelder:

- Methoden und Werkzeuge der Software-Entwicklung
- Engineering-Prozesse
- Entwicklungsstandards (Leistungsfähigkeit, Sicherheit, Verfügbarkeit, Innovation)
- Projektplanung und -management
- Qualitätsstandards
- Moderation
- Konfliktbewältigung

³ Die Kompetenzfelder werden in der nachfolgenden Auflistung jeweils durch ein zusammenfassendes Stichwort benannt. Da die Weiterbildung zum Spezialisten auf die erfolgreiche Bewältigung zunehmend offener beruflicher Handlungssituationen sowie ganzheitlichen Kompetenzerwerb abzielt, bildet der Kompetenzerwerb einen integralen Bestandteil der Arbeits- und Weiterbildungsprozesse und lässt sich nur im Zusammenhang mit diesen operationalisieren (vgl. dazu die Abschnitte „Kompetenzfelder“ in den Kapiteln 3.1.4ff).

⁴ Jeder Spezialist muss in den in diesem Abschnitt genannten „weichen“ Kompetenzfeldern wie „Kommunikation, Präsentation“, „Konflikterkennung, -lösung“ usw. ein Niveau erreichen, das über dem einer Fachkraft liegt. Das heißt, er muss auch in diesen Feldern zu eigenständigem Handeln in der Lage sein und zum Erreichen des Ziels in dem jeweiligen Feld ggf. über den Rahmen bekannter Verfahren und Lösungen hinausgehen können.

Routiniert zu beherrschende, profilspezifische Kompetenzfelder:

- Design-Methoden, -Strategien,
- Systemarchitekturen
- Systemintegration und -anpassung
- Bussysteme, Protokolle und Schnittstellen
- Datenbanken, Betriebssysteme
- Teststrategien, -design -planung
- Testmanagement, Testautomatisierung, Testumgebungen
- Fehlermanagement
- statistische Verfahren
- Programmier- und Skriptsprachen
- Branchenüberblick

2.4 Qualifikationserfordernisse

Im Regelfall wird ein hinreichendes Qualifikationsniveau auf der Basis einschlägiger Berufsausbildung oder Berufserfahrung vorausgesetzt. Falls noch fachliches Know-how fehlt, existiert eine ganze Reihe Schulungs- und Qualifizierungsmaßnahmen, in denen die Grundlagen, Begriffe und Techniken des Software-Tests vermittelt werden, z. B. akkreditierte Trainings nach dem Europäischen Certified Tester Standard [ISTQB.org].

2.5 Einordnung ins System und Karrierepfade

Das neue IT-Weiterbildungssystem gibt auf Basis der vier neuen IT-Ausbildungsberufe drei Ebenen für die Weiterqualifizierung vor: Spezialisten, wie der IT Test Coordinator einer ist, operative und strategische Professionals. Auf der Ebene der Spezialisten existiert eine Reihe verwandter Profile und selbstverständlich kann sich auch der IT Test Coordinator zu einem Professional weiterqualifizieren.

Verwandte Profile

Der IT Test Coordinator weist eine Verwandtschaft zu zwei Profilgruppen auf. Einmal zu den anderen Coordinator/Entwicklungsprozessbegleitern, insbesondere zum Quality Management Coordinator und zum Project Coordinator. Die Tätigkeiten des IT Test Coordinator gleichen dabei den Tätigkeiten eines Project Coordinator mit der Einschränkung, dass sich die Koordinierungsaufgaben auf ein „Testprojekt“ beziehen. Die andere Profilgruppe sind die Entwickler, da ein Schwerpunkt der Tätigkeiten die Erstellung der Testtreiber darstellt.

Aufstiegsqualifizierung

Die am nächsten liegende Aufstiegsmöglichkeit stellt der IT Systems Engineers dar, der ähnlich wie der Test Coordinator einen Überblick über den gesamten Entwicklungsprozess haben muss. Ebenfalls möglich ist der Aufstieg zum IT Business Manager als Projektmanager sowie IT Business Consultant, bspw. für externe Tests. Eine weitere Entwicklungsmöglichkeit für den IT Test Coordinator außerhalb des IT-Weiterbildungssystems stellt der Berater bzgl. Testprozessoptimierung dar.

3 Referenzprozess

Der Referenzprozess des IT Test Coordinator lässt sich in sechs verschiedene Arbeitsgebiete gliedern. Die Arbeitsgebiete sind:

- Erstellung des Testplans (Planungsphase)
- Erstellung und Durchführung von Modultests mit dem White-Box-Verfahren
- Erstellung und Durchführung von Modultests mit dem Black-Box-Verfahren
- Erstellung und Durchführung von Integrationstests
- Erstellung und Mitwirken bei System-/Last-/Performanztests
- Auswertung des Testprotokolls/Erstellung Testbericht

Diese Arbeitsgebiete sind Bestandteile unterschiedlicher Phasen des Software-Entwicklungsprozesses. Innerhalb der Erstellung und Durchführung der einzelnen Tests auf den Teststufen (Modul, Integration, System) wiederholen sich die Arbeitsschritte.

3.1 Kerntätigkeiten des IT Test Coordinator

Die Durchführung der Modul-, Integrations-, Systemtests als Kern des Referenzprozesses des IT Test Coordinator besteht - kurz zusammengefasst - aus folgenden ineinander greifenden Teilen, die in den Teilprozessen immer wieder auftreten:

- Test planen/instanzieren
- Testspezifikation erstellen
- Erstellen einer Testumgebung/Testautomatisierung
- Testdurchführung und Protokollierung
- Auswertung des Testprotokolls
- Erstellen des zusammenfassenden Testberichts

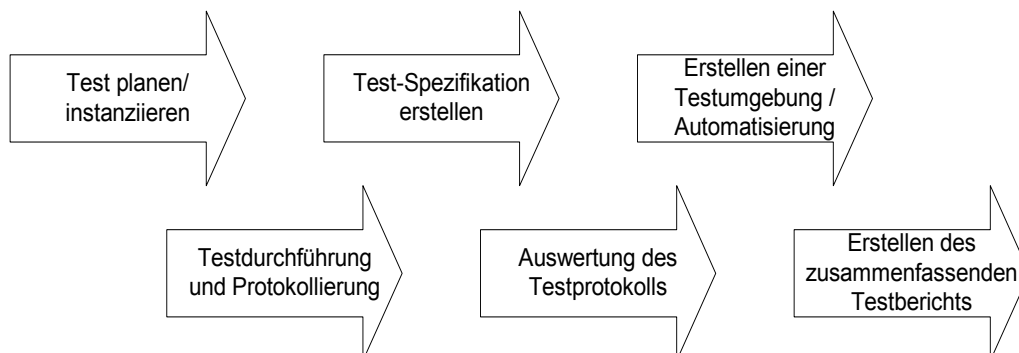


Abbildung F: Zusammenfassung der Kerntätigkeiten des IT Test Coordinator.

Die Abbildung zeigt die zeitliche Reihenfolge der einzelnen Arbeitsschritte eines IT Test Coordinator innerhalb einer Teststufe. Die Arbeitsschritte sind angelehnt an den Testprozess gemäß der Darstellung im [BS7925-1]. Dabei unterscheidet man folgende Teststufen:

- Modultest als White Box Test: Test einer Software-Einheit (Modul) im White-Box-Verfahren
- Modultest als Black Box Test: Test einer Software-Einheit (Modul, Komponente) im Black-Box-Verfahren
- Integrationstest: Test mit dem Ziel, Fehler in Schnittstellen und im Zusammenspiel zwischen integrierten Komponenten zu finden
- Systemtest: Test eines integrierten Systems, um sicherzustellen, dass es spezifizierte Anforderungen erfüllt

Der Aufbau der im Testprozess zu erstellenden Testdokumente wird im IEEE829-Standard[IEEE 829] definiert. Der Testprozess als Bestandteil eines Software-Entwicklungsprozesses wird im V-Modell beschrieben. Auf der Basis dieser beiden Standards wurde das Spezialistenprofil für den IT Test Coordinator definiert. Dabei beschreibt das Spezialistenprofil die Arbeitsschritte in den Teilprozessen der Referenzprozesse aus Sicht eines IT Test Coordinator.

Die Arbeitsschritte innerhalb der Teststufen werden in den folgenden Teilprozessen ausführlich dargestellt.

Der Referenzprozess gibt die gesamten Aktivitäten des IT Test Coordinator auf hohem Abstraktionsniveau wieder und ermöglicht so einen Überblick.

Mit den Teilprozessen wird in den Referenzprozess hineingezoomt. Die Teilprozesse entsprechen damit in etwa der Abbildung von Arbeitsprozessen; sie stellen einen konkreten Tätigkeitsverlauf, einschließlich auslösendem Ereignis und Ergebnis, dar.

Die zur Durchführung der Teilprozesse notwendigen Tätigkeiten und Kompetenzfelder werden jeweils in einem separaten Abschnitt aufgelistet.

Das Praxisprojekt der Firma imbus dient als Beispiel zur Konkretisierung und Veranschaulichung des Referenzprozesses. Es ist ein echtes, bereits durchgeführtes Projekt, auf dessen Grundlage die hier dargestellten Referenz- und Teilprozesse entwickelt wurden.

Literatur:

- [BS7925-1]: Britischer Standard für Software Testing Vocabulary, Computer Society Specialist Interest Group in Software Testing (BCS SIGIST).
- [IEEE 829] ANSI/IEEE Std 829-1983, IEEE Standard for Software Test Documentation, Institute of Electrical and Electronics Engineers, Inc., August 1983.
- [Spillner/Linz] Eine Einführung in den Testprozess gibt das Buch „Basiswissen Softwaretest“ von Spillner und Linz.
- [V-Modell], Entwicklungsstandards für IT-Systeme des Bundes, <http://www.v-modell.iabg.de>, 1993.

3.1.1 Referenzprozess IT Test Coordinator

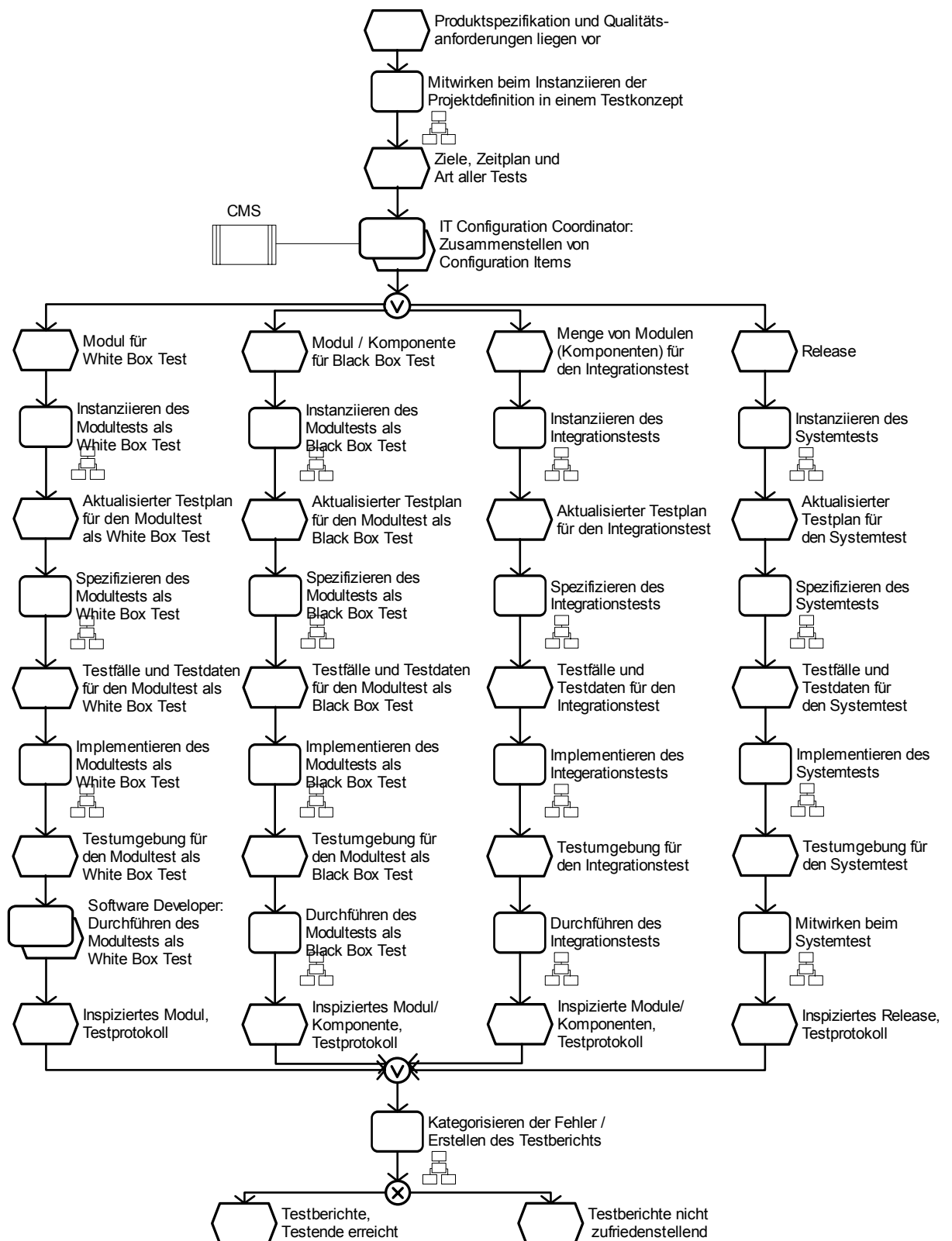


Abbildung G: Referenzprozess des IT Test Coordinator.

Der IT-Test-Coordinator-Referenzprozess ist typisch für ein IT-Test-Coordinator-Profil: Der Referenzprozess ist im eigentlichen Sinne kein kontinuierlicher Prozess, sondern besteht aus einzelnen Teilprozessen, die sich an die einzelnen Entwicklungsschritte (Modul, Integration, System) im Software-Entwicklungsprozess anschließen. Der Referenzprozess beginnt mit der Erstellung des Testkonzepts innerhalb der Projektplanung. Nach der Festlegung des Testkonzepts geht es mit der Instanziierung des Testkonzepts in der jeweiligen Teststufe weiter zur Spezifizierung des Tests in der Testspezifikation und der Implementierung der Testumgebung. Als Teststufe werden Modul-, Integrations- und Systemtests vorbereitet und durchgeführt. Innerhalb der Durchführung der verschiedenen Tests wird das Testprotokoll erstellt, das zum Abschluss jeder Teststufe ausgewertet und in einem Testbericht zusammengefasst wird.

Dieser Prozess läuft in kleinen wie in großen Projekten ab, in denen Software-Entwicklung eine Rolle spielt. Allerdings wird nicht in jedem Projekt jeder Teilprozess den gleichen Umfang und die gleiche Komplexität haben. So werden in großen Projekten sicherlich mehrere IT Test Coordinator zusammenarbeiten und es wird insbesondere bei den Modultests eine enge Zusammenarbeit mit den Software-Entwicklern geben. In kleinen Projekten hingegen kann der IT Test Coordinator u. U. alle Teststufen allein durchführen.

3.1.2 Die Beispielprojekte

Die Firma imbus ist Spezialist für Software-Qualitätssicherung und -Test. Sie wurde von einem Software-Hersteller beauftragt, verschiedene Testaufgaben für ein neues Produkt zu übernehmen. Die Testaufgaben werden mittlerweile bereits über mehrere Jahre über mehrere Produktreleases von der Firma imbus betreut. Mit der Erledigung der Aufgaben ist im imbus-Testlabor ein Team von 5-10 Testern betraut.

Die einzelnen Testaufgaben lassen sich verschiedenen Teststufen zuordnen. Wie bereits beschrieben, wird in jeder Teststufe dasselbe Vorgehen angewandt. Es kommt jeweils der Testprozess entsprechend Spillner/Linz zur Anwendung.

Die im Folgenden beschriebenen Beispielprojekte dienen als Praxisprojekte für alle Tests, die im Black-Box-Verfahren getestet werden. Durch die Vergabe der Tests an ein externes Unternehmen gibt es Abweichungen bei den Kommunikationswegen. Die Kommunikation wird im Wesentlichen zwischen den Projektleitern (intern, extern) gebündelt. Diese Spezifika sind im Referenzprozess aber nicht berücksichtigt worden.

Die folgenden Projekte sind in Bezug auf die Einordnung in Teststufen und in der Zielsetzung verschieden:

3.1.2.1 Beispielprojekt für den Modultest nach dem Black-Box-Verfahren

Die verantwortliche Entwicklungsabteilung beauftragte die Firma imbus mit der Übernahme von Modultests für ein größeres Modul. Zielsetzung war die Erstellung einer Testspezifikation und Testautomatisierung. Die erstellte Testautomatisierung wurde nach Abnahme durch den Auftraggeber in den Build-Prozess integriert. Die Auswertung der beim Build durchgeführten Regressionstests wurde vom Kunden vorgenommen.

3.1.2.2 Beispielprojekt für den Integrationstest

Im zweiten Beispielprojekt wurde imbus beauftragt, funktionale Tests für eine Komponente zu entwickeln. Innerhalb von drei Jahren wurde in Vorgängerprojekten durch imbus eine umfangreiche Testsuite aufgebaut. Es existieren ca. 900 Testfälle, von denen etwa 300 automatisiert sind.

Die Tests wurden auf Basis der GUI betrieben. Neue Versionen der Komponente wurden an imbus mit dem Gesamtsystem geliefert. Die Tests haben somit Systemtestcharakter. Jedoch konzentrierten sich die Tests ausschließlich auf die Funktionalität der Komponente. Ziel ist eine funktionale Abdeckung. Die Funktionalität der Komponente wurde gegen das Design-Dokument „Funktionale Spezifikation“ und gegen diverse Style Guides (z. B. Windows Style Guide) getestet.

Innerhalb eines neuen Produktreleases wurden vom Auftraggeber Änderungen und Bugfixes an der zu testenden Komponente vorgenommen. Die Testaufgabe innerhalb eines Produktreleases besteht in Folgendem:

- Wiederholung einer Anzahl von Tests um zu prüfen, ob die geplanten Änderungen keine unbeabsichtigten Änderungen an der bestehenden Funktionalität mit sich gebracht haben (Regressionstests)
- Spezifikation neuer Testfälle zur Überprüfung der neuen Funktionalität
- Automatisierung neuer Testfälle
- Durchführung neuer Testfälle

3.1.2.3 Beispielprojekt für den Systemtest – Schwerpunkt Installation und Lizenzierung

Im dritten Teilprojekt wirkt imbus am Systemtest für ein komplettes Produkt mit. Das Produkt besteht aus einer Anzahl von Komponenten. Die einzelnen Komponenten werden als jeweils bereits getestet betrachtet.

Für imbus besteht das Ziel darin, das Produkt im Black-Box-Verfahren aus der Sicht des Endanwenders zu testen. Folgende Testschwerpunkte wurden mit dem Auftraggeber vereinbart:

- Test der Installationsroutinen
- Test der Migration von Daten aus Vorversionen
- funktionale Regressionstests für das Produkt

3.1.2.4 Überblick über die Beispielprojekte

Charakteristik		Modultest	Integrationstest	Systemtest
Charakterisierung der Teststufe	Testobjekt	mit Java implementiertes Modul	neue Version der Komponente eingebettet ins Produkt Identifikation der Komponente über Features/Funktionalität	komplettes Produkt speziell Installation und Lizenzierung
	Testziele	Erstellen einer Testspezifikation und Testautomatisierung funktionale Abdeckung gegen die Schnittstellenbeschreibung des Moduls und die funktionale Spezifikation Integration in den Build	Regressionstests Anpassung von Testspezifikation und Testautomatisierung für neue Features	finaler Test aus der Sicht des Endanwenders funktionaler Test der Installation und Lizenzierung des Produkts Test gegen Requirements und Produktdesign
	Points of Control (POC)	Interface-Klassen und -Methoden	Komponenteninterface über GUI	GUI, Online-Dokumentation
	Points of Observation (POO)	wie POC	wie POC	wie POC
	Points of Simulation (POS)	Simulation der Datenbankanbindung	keine Simulation	keine Simulation

	Teilaktivität	Modultest	Integrationstest	Systemtest
Inhalte der Teilaktivitäten im Testprozess	Testplanung/Instanziierung	<p>Zu Beginn eines jeden Testzyklus' wird die Testplanung (→ Dokument Testplan) angepasst. Insbesondere wird die Planung in folgenden Situationen angepasst:</p> <p>Testfortschritt in vorigen Testzyklen entspricht nicht der Planung</p> <p>Änderungen in der Release-Planung durch die Entwicklung, z. B.</p> <p>Feature/Komponente wurde nicht rechtzeitig fertig und kann erst im nächsten Zyklus getestet werden</p> <p>neue Feature/Komponenten werden im Gegensatz zur ursprünglichen Planung integriert</p> <p>Qualitätsproblem für Bereiche der Software (Instabilitäten, hohe Anzahl von Fehlermeldungen)</p>		
	Testspezifikation	<p>Test aller Schnittstellenmethoden</p> <p>Variation der Parameter mit Äquivalenzklassenmethode</p>	<p>Use Cases für Komponente</p> <p>Parametrierung der Use Cases über Äquivalenzklassenmethode</p> <p>Parametrierung der Eingaben in Masken über Äquivalenzklassenmethode</p>	<p>Tests gegen Requirements</p> <p>Use Cases für Produkt</p> <p>die Äquivalenzklassenmethode wird nur vereinzelt angewandt</p>
	Testimplementierung	<p>Java-basierte Testimplementierung mit J-Unit</p> <p>(→ http://www.junit.org)</p>	<p>ca. 50 % der Tests sind mithilfe des Testautomatisierungstools (sog. Testroboter bzw. Capture and Replay Tools) automatisiert</p> <p>(http://www.imbus.de/testlabor/tool-list.html)</p>	<p>ca. 10-15 % der Tests sind mithilfe eines Testautomatisierungstools automatisiert</p>
	Testdurchführung	<p>vor Release: Durchführung der Tests in Verbindung mit T-Aut.</p> <p>nach Release: Integration der Tests in den Build</p>	<p>Durchführung der automatisierten Tests</p> <p>manuelle Verifikation von Bugfixes</p> <p>vereinzelte manuelle Testdurchführung</p>	<p>manuelle Testdurchführung komplexer Szenarien (z. B. Installation eines Produkts)</p> <p>Testautomatisierung wirkt nur unterstützend</p>

3.1.3 Prozesskompass IT Test Coordinator

Dieser Prozesskompass enthält die im Referenzprozess dargestellten Teilprozesse:

- Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept
- Instanzieren des Modultests als White Box Test
- Spezifizieren des Modultests als White Box Test
- Implementieren des Modultests als White Box Test
- Instanzieren des Modultests als Black Box Test
- Spezifizieren des Modultests als Black Box Test
- Implementieren des Modultests als Black Box Test
- Durchführen des Modultests als Black Box Test
- Instanzieren des Integrationstests
- Spezifizieren des Integrationstests
- Implementieren des Integrationstests
- Durchführen der Integrationstests
- Instanzieren des Systemtests
- Spezifizieren des Systemtests
- Implementieren des Systemtests
- Mitwirken beim Systemtest
- Kategorisieren der Fehler/Erstellen des Testberichts

3.1.4 Teilprozesse des IT Test Coordinator

Die Teilprozesse geben die Tätigkeiten während der Erstellung und Durchführung der Tests auf den verschiedenen Teststufen innerhalb eines Entwicklungsprozesses ausführlich und detailliert wieder. Sie entsprechen so den Tätigkeiten eines IT Test Coordinator in einem realen Projekt, welches als Grundlage für den Referenz- und die Teilprozesse gedient hat und als Beispiel zur Veranschaulichung beschrieben wird.

Sicherlich werden nicht in jedem Projekt alle diese Teilprozesse vorkommen. Insbesondere Prozesse wie das Instanzieren der Projektdefinition in einem Testkonzept oder das Spezifizieren der Anforderung für Modultests im White-Box-Verfahren sind sehr von der konkreten Einbettung des Projekts abhängig. Auch die Tätigkeiten bei dem Modultest im White-Box-Verfahren werden, wenn überhaupt, in enger Kooperation mit den Software-Entwicklern durchgeführt. Ein IT Test Coordinator auf der Spezialistenebene sollte allerdings alle diese Prozesse beherrschen. Die Betonung liegt dennoch auf den Prozessen, die für die Durchführung Modul-, Integrations- und Systemtest im Black-Box-Verfahren zentral sind, angefangen vom Identifizieren der Testfälle bis hin zum Mitwirken bei Systemtests und Auswerten der Testprotokolle.

3.1.4.1 Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept

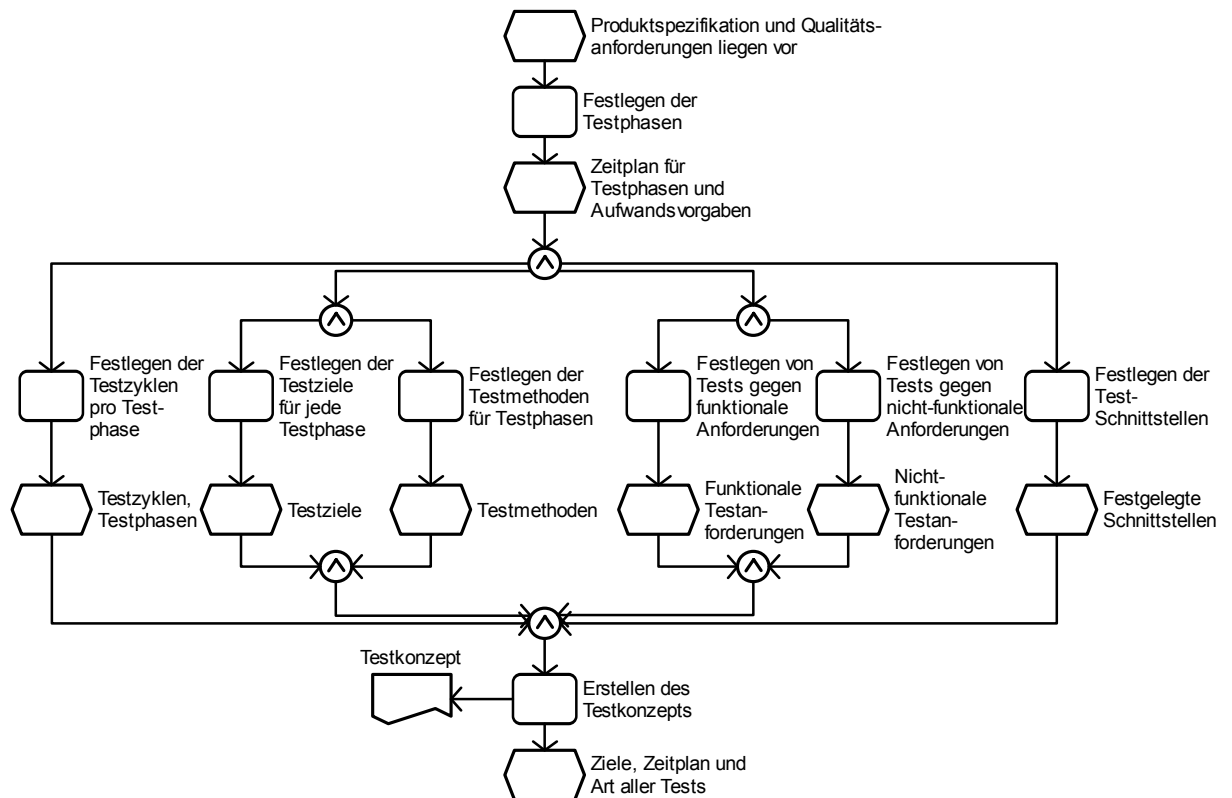


Abbildung 1: Mitwirken beim Instanzieren der Projektdefinition in einem Testplan.

Dieser Teilprozess ist Bestandteil der Planungsphase eines Software-Entwicklungsprozesses. In dieser Projektphase wird der Projektplan mit dem Testkonzept und dem Konfigurationsmanagement-Plan erstellt. Im Testkonzept werden die Art der Tests (funktionale, nicht funktionale), die Teststufen und die generell anzuwendenden Testmethoden festgelegt. Da der Testplan in erster Linie die zeitlichen Aspekte (Teststufen mit Testzyklen) widerspiegelt, werden im Testkonzept die strategischen Aspekte wie Testziel, Vorgehensweise, anzuwendende Testmethoden und die Schnittstellen der zu testenden Komponenten beschrieben.

3.1.4.1.1 Tätigkeiten: Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept

- Festlegen der Teststufen, ggf. in Zusammenarbeit mit dem Projektleiter
- Festlegen der Testzyklen pro Teststufe
- Festlegen der Testziele für jede Teststufe
- Festlegen der Testmethode; ggf. in Zusammenarbeit mit dem Qualitätssicherungsbeauftragten
- Festlegen von Tests gegen funktionale Anforderungen, ggf. in Zusammenarbeit mit dem Projektleiter
- Festlegen von Tests gegen nicht funktionale Anforderungen, ggf. in Zusammenarbeit mit dem Projektleiter
- Festlegen der Testschnittstellen

- Erstellen des Testplans/Testkonzepts, ggf. in Zusammenarbeit mit dem Projektleiter und Qualitätssicherungsbeauftragten

3.1.4.1.2 Kompetenzfelder: Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept

Fähigkeiten/Fertigkeiten

- (nicht) funktionale Anforderungen in Testmethoden, Testziele umsetzen können
- passende Testmethoden für die Teststufen und Testziele identifizieren können
- Testziele für die einzelnen Teststufen identifizieren können
- Schätzung der Aufwände und Ressourcen zu den einzelnen Teststufen vornehmen können
- Schätzung der Aufwände für die Testzyklen vornehmen können
- kommunizieren können
- zusammenarbeiten können
- Kompromisse eingehen können
- sich durchsetzen können
- präsentieren können
- Wesentliches von Unwesentlichem unterscheiden können
- Überblick bewahren können

Wissen

- Testziele
- Grundkenntnisse Projektmanagement

Werkzeuge/Methoden

- Testmethoden für funktionale Anforderungen für verschiedene Teststufen
- Testmethoden für nicht funktionale Anforderungen für verschiedene Teststufen
- QM-System zur Unterstützung der Festlegung der Testziele und Testmethoden
- Planungssysteme für das Software-Projekt

3.1.4.1.3 Beispiel: Mitwirken beim Instanzieren der Projektdefinition in einem Testkonzept

Bei dem betrachteten Beispielprojekt handelte es sich um einen neuen Produktrelease einer bestehenden Software. Nicht auf alle der in dieser Phase betrachteten Fragen musste eine grundlegende Antwort gegeben werden. Die Aufteilung der Verantwortlichkeiten sowie die Verwendung von Prozessen konnte aus den vorigen Projekten übernommen werden. Einzig die Definition von Testzielen und Teststrategie für neue Funktionalität und bestehende Funktionalität wurde angepasst. Im vorliegenden Projekt wurde das Testkonzept vom Projektleiter des Auftraggebers erstellt. Der imbus-Projektleiter unterstützte diese Aktivität durch das Vorschlagen von Testaktivitäten sowie die Abschätzung der Testaufwände.

Für neue Projekte wird bei imbus typischerweise ein Testkonzept mit folgender Gliederung erstellt:

0 EINFÜHRUNG	5
0.1 REFERENZIERTE DOKUMENTE	5
1 ZU TESTENDES SYSTEM UND TESTOBJEKTE	6
1.1 ÜBERBLICK ÜBER DAS ZU TESTENDE SYSTEM	6
1.2 TESTOBJEKT 1	6
1.3 TESTOBJEKT I	6
1.4 SONSTIGE TESTOBJEKTE	6
1.5 KOMPONENTEN DIE NICHT GETESTET WERDEN	7
1.5.1 DB-System	7
1.5.2 Komponente 2	7
2 ORGANISATION DER TESTS	8
2.1 EINBINDUNG IN DAS GESAMTPROJEKT	8
2.2 DIE TESTINSTANZEN	8
2.3 TESTTEAM	8
2.4 BERICHTSWESEN	8
3 RISIKOBEACHTUNG UND PRIORISIERUNG	9
3.1 PRIORISIERUNG	9
3.2 RISIKOBEACHTUNG	9
4 TESTZIELE, TESTSTRATEGIE UND TESTTHEMEN	10
4.1 TESTZIELE (DER TESTINSTANZ)	10
4.2 TESTSTRATEGIE	10
4.2.1 Testthema 1	10
4.2.2 Testthema 2 ...	10
4.2.3 Spezielle Testthemen für Testobjekt 1	10
4.2.4 Spezielle Testthemen für Testobjekt i	10
4.2.5 Leistungsmerkmale die nicht getestet werden	10
5 VERFAHREN, METHODEN UND WERKZEUGE	11
5.1 PROZESS	11
5.1.1 Überblick über den Testprozeß	11
5.1.2 Testdurchführung	11
5.2 SOFTWAREÜBERGABEVERFAHREN	11
5.3 KONFIGURATIONSMANAGEMENT	11
5.3.1 Für Testobjekte und Entwicklungsdokumente	11
5.3.2 Für Tests und Testdokumente	11
5.3.3 Konfigurationsmanagement-Tool	11
5.4 FEHLERMANAGEMENT	11
5.4.1 Fehlermeldeverfahren	11
5.4.2 Fehlerstatusmodell	12
5.4.3 Fehlerdatenbank (FDB)	12
5.5 ÄNDERUNGSMANAGEMENT	12
5.6 TESTUMGEBUNG	12
5.6.1 Testplattform(en)	12
5.6.2 Tester-Arbeitsplätze	13
5.6.3 Tools zur Testautomatisierung	13
5.6.4 Entwicklungsumgebung	13
5.6.5 Sonstige Werkzeuge	13
5.6.6 Tool-Administration	13
6 TESTMETRIKEN	14
6.1.1 Fehleranzahl	14
6.1.2 Fehleraten	14
6.1.3 Test-Fortschritt	14
6.1.4 Produktivität	14
6.1.5 Produktqualität	14

Abbildung 2: Inhaltsverzeichnis eines Testkonzepts.

Zusätzlich entsteht in dieser Phase bereits eine Plan, der die die Anzahl der geplanten Testzyklen und die Aufteilung der Testaktivitäten auf die einzelnen Testzyklen enthält. Als Basis dafür dient die Liste der Features, die zum Zeitpunkt der erstellten Zwischenversion bereits integriert werden sollen.

Sample Testplan Version	Test cycle TC11	Efforts in Engineering		Progress in	Test cycle TC2	Efforts in Engineering		Progress in
		Days (ED)	%			Days (ED)	%	
Planned cycle dates	7/1/2002 - 7/22/2002				7/23/2002 - 8/5/2002			
Test cycle duration	16				10			
Main emphasis	Feature 1 Complete Feature 2 First				Feature 2 Complete Feature 3 Complete			
Prepare test cycle								
Prepare test cycle		1				1		
Acceptance Tests (test cases marked with Prio 0 in test spec)								
Acceptance Tests	All test cases	10	0%		All test cases	10	0%	
02 Installation and Licensing								
02.1 Installation	All test cases	5	0%		Only high priority test cases	3	0%	
02.2 Deinstallation	All test cases	3	0%		Only high priority test cases	2	0%	
02.3 Licensing	Only high priority test cases	1	0%		none			
03 Platform tests								
03.1 Client platforms	none				none			
03.2 Server platforms	Linux Platforms	0,5	0%		none			
04 Functional Tests								
04.1 Feature1	All test cases	5	0%		Only high priority test cases	5	0%	
04.2 Feature2	Test specification	5	0%		All test cases	5	0%	
04.3 Feature3	none				All test cases	5	0%	
Defect retest								
Defect retest	All	2	0%		All	3	0%	
Total		32,5				34		

Abbildung 3: Testplan mit Testzyklen.

3.1.4.2 Instanzieren des Modultests (als White Box Test)

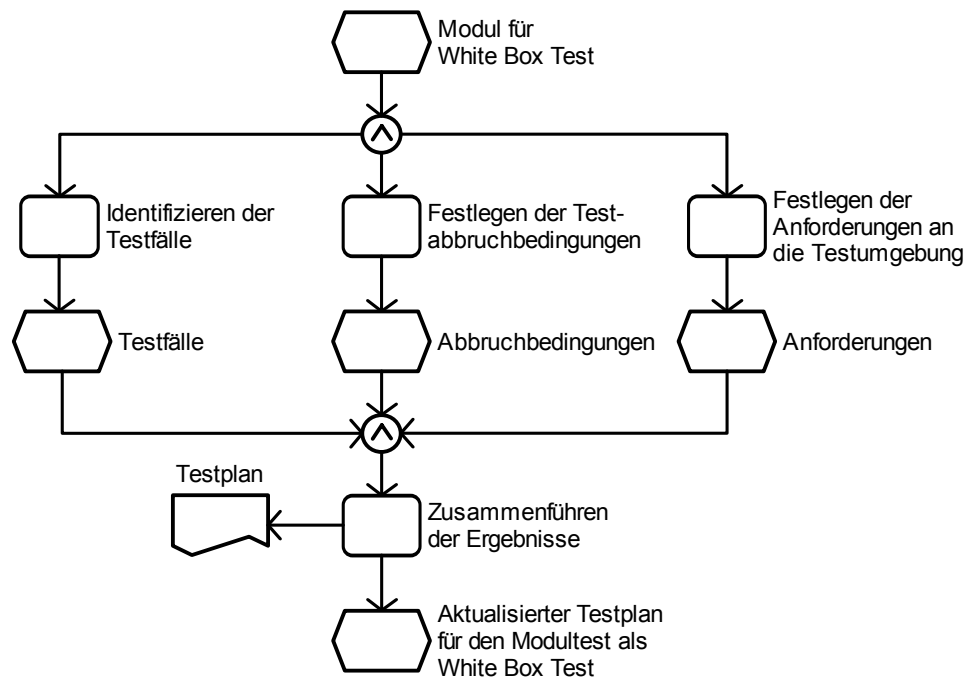


Abbildung 4: Instanzieren des Modultests (als White Box Test).

Diese Aufgabe lässt sich in die drei Teilaufgaben „Identifizieren der Testfälle“, „Festlegen der Testabbruchbedingungen“ und „Festlegen der Anforderungen an die Testumgebung“ zerlegen. Die Aufgabe „Zusammenführen der Ergebnisse“ liefert die Testanforderungen und die Abbruchbedingungen als Ergebnis in einem Testplan.

Bei einem White Box Test werden auf der Basis des Quelltextes des zu testenden Objekts die Testfälle entwickelt. Als mögliche Methoden zur Identifizierung der Testfälle dienen z. B. verschiedene Code-Überdeckungsverfahren.

3.1.4.2.1 Tätigkeiten: Instanzieren des Modultests (als White Box Test)

- Identifizieren der Testfälle für den Modultest auf der Basis des Testobjekts
- Festlegen der Testabbruchbedingungen gemäß der Testplanung oder den Richtlinien aus dem QM-System
- Festlegen der Fehlerklassen
- Festlegen der Anforderungen an die Testumgebung
- Zusammenführen der Ergebnisse des Modultests in einem Testkonzept als Ergebnisdokument dieses Teilprozesses

3.1.4.2.2 Kompetenzfelder: Instanzieren des Modultests (als White Box Test)

Fähigkeiten/Fertigkeiten

- Code-Überdeckungsverfahren anwenden können
- Testabbruchbedingungen definieren können
- Anforderungen an Testumgebung festlegen können

- Testkonzept erstellen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- logisch denken können
- abstrakt denken können
- Zusammenhänge erkennen können
- Kreativität im Generieren von Testfällen

Wissen

- Fehlerklassen
- Basiswissen zum Konfigurationsmanagement

Werkzeuge/Methoden

- Code-Überdeckungsmethoden
- Code-Metriken
- QM-System
- Entwicklungswerkzeuge

3.1.4.2.3 Beispiel: Instanzieren des Modultests (als White Box Test)

Diese Phase dient als Planungsphase für den Modultest. In dem in Abschnitt 3.1.4.1 erstellten Testkonzept wurden Qualitätsziele und Teststrategie festgelegt.

Ein Beispiel für die konkreten Tätigkeiten eines IT Test Coordinator beim Instanzieren der Tests wird im Teilprozess 3.1.4.13 „Instanzieren des Systemtests“ beschrieben. Da der Prozessablauf in den verschiedenen Teststufen immer der gleiche ist, wird hier auf ein Beispiel verzichtet.

3.1.4.3 Spezifizieren des Modultests (als White Box Test)

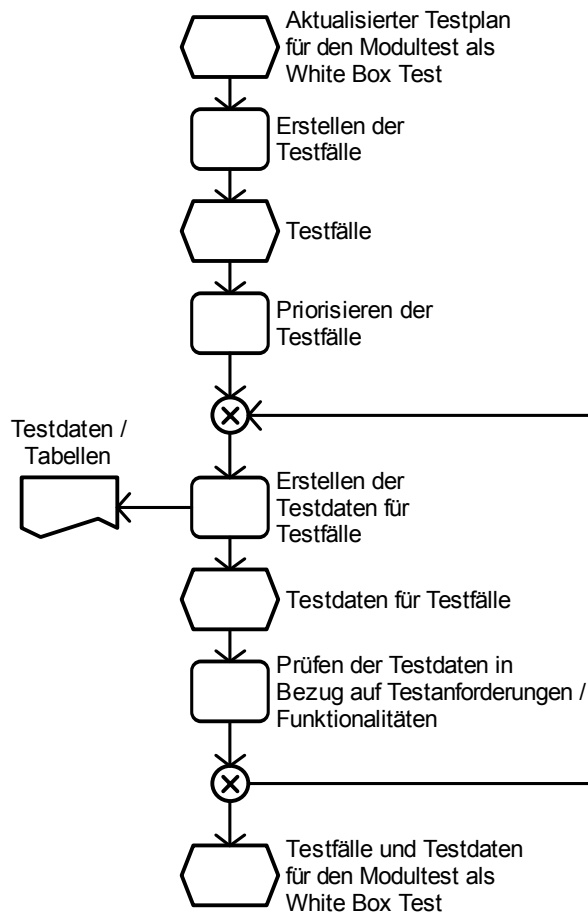


Abbildung 5: Spezifizieren des Modultests (als White Box Test).

In diesem Teilprozess werden die Testdaten für die einzelnen Testfälle erstellt. Anschließend werden die Testfälle priorisiert. Zum Schluss wird zum Beispiel durch Reviews überprüft, ob Testfälle und Testdaten die gewünschten Testanforderungen wie etwa die Testabdeckung erreichen.

3.1.4.3.1 Tätigkeiten: Spezifizieren des Modultests (als White Box Test)

- Erstellen der Testfälle
- Priorisieren der Testfälle; trotz Testplanung mit Zeit und Personalplanung ist eine Priorisierung der Testfälle sinnvoll, um die wichtigsten Testfälle (Prio1) auf jeden Fall bei einer Zeitüberschreitung getestet zu haben
- Erstellen der Testdaten für Testfälle
- Prüfen der Testdaten in Bezug auf die Testanforderungen/Funktionalitäten

3.1.4.3.2 Kompetenzfelder: Spezifizieren des Modultests (als White Box Test)

Fähigkeiten/Fertigkeiten

- Testdaten zu Testfällen erstellen können
- Testfälle erstellen können

- Testfälle priorisieren können
- Überdeckung der Testfälle mit Testanforderungen überprüfen können
- Prioritäten setzen können
- Sachverhalte in unterschiedliche Kontext einordnen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- kreativ denken können
- logisch denken können
- abstrakt denken können
- methodisch denken können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebungen
- Software-Entwicklungsprozesse

Werkzeuge/Methoden

- Testmethoden für Modultests im White-Box-Verfahren
- QM-System

3.1.4.3.3 Beispiel: Spezifizieren des Modultests (als White Box Test)

Die Testspezifikation wurde anhand der festgelegten Teststrategie aus 3.1.4.2 erstellt. Ein Beispiel für die konkreten Tätigkeiten eines IT Test Coordinator beim Spezifizieren der Anforderungen für die Tests wird im Teilprozess 3.1.4.14 „Spezifizieren des Systemtests“ beschrieben. Da der Prozessablauf in den verschiedenen Teststufen immer der gleiche ist, wird hier auf ein Beispiel verzichtet.

3.1.4.4 Implementieren des Modultests (als White Box Test)

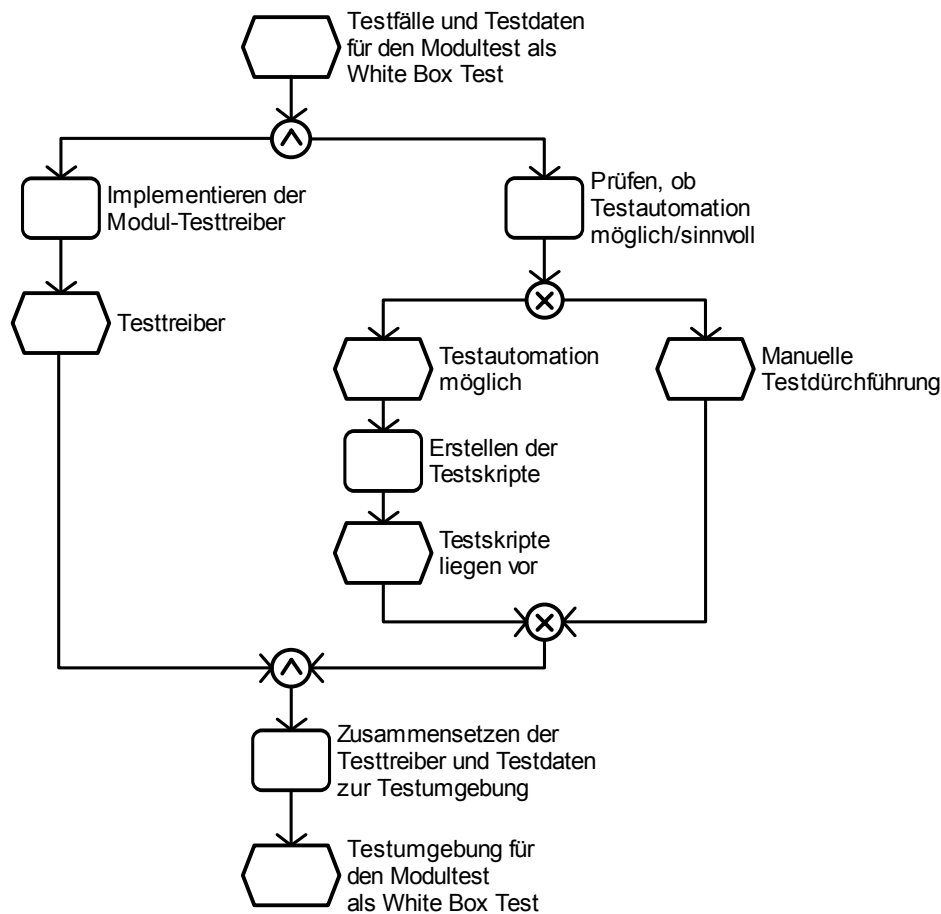


Abbildung 6: Implementieren der Modultests (als White Box Test).

In diesem Teilprozess wird die Entwicklung der Testumgebung und die Überprüfung auf Testautomation beschrieben. Dabei wird die Testumgebung mithilfe der Schritte „Implementieren der Testtreiber“ und „Zusammensetzen der Testtreiber, Testobjekte, Testdaten zur Testumgebung“ erstellt. Parallel hierzu wird überprüft, ob eine Testautomation sinnvoll ist und für die entsprechenden Testfälle werden dann die Testskripte erstellt.

3.1.4.4.1 Tätigkeiten: Implementieren der Modultests (als White Box Test)

- Implementieren der Modultesttreiber in Zusammenarbeit mit für das Testobjekt verantwortlichen Software-Entwicklern
- Zusammensetzen der Testtreiber und Testdaten des Testobjekts zur Modul-Testumgebung
- Prüfen, ob Testautomation möglich/sinnvoll ist
- Erstellen der Testskripte; in Abhängigkeit vom eingesetzten Testwerkzeug werden Testskripte für die Testfälle erstellt

3.1.4.4.2 Kompetenzfelder: Implementieren der Modultests (als White Box Test)

Fähigkeiten/Fertigkeiten

- Testtreiber implementieren können
- Testumgebung erstellen können
- Testskripte erstellen können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebungen
- Software-Entwicklungsprozess

Werkzeuge/Methoden

- Code-Überdeckungsmethoden und Werkzeuge zur Messung der Code-Überdeckung
- Code Metriken
- Testmethoden für Modultests im White-Box-Verfahren
- Programmierumgebungen/Compiler/Interpreter
- Testautomationswerkzeuge
- QM-System

3.1.4.4.3 Beispiel: Implementieren der Modultests (als White Box Test)

Ein Beispiel für die konkreten Tätigkeiten eines IT Test Coordinator bei der Implementierung einer Testumgebung wird im Teilprozess „Implementieren des Systemtests“ beschrieben. Da der Prozessablauf in den verschiedenen Teststufen immer der gleiche ist, wird hier auf ein Beispiel verzichtet.

3.1.4.5 Instanzieren des Modultests (als Black Box Test)

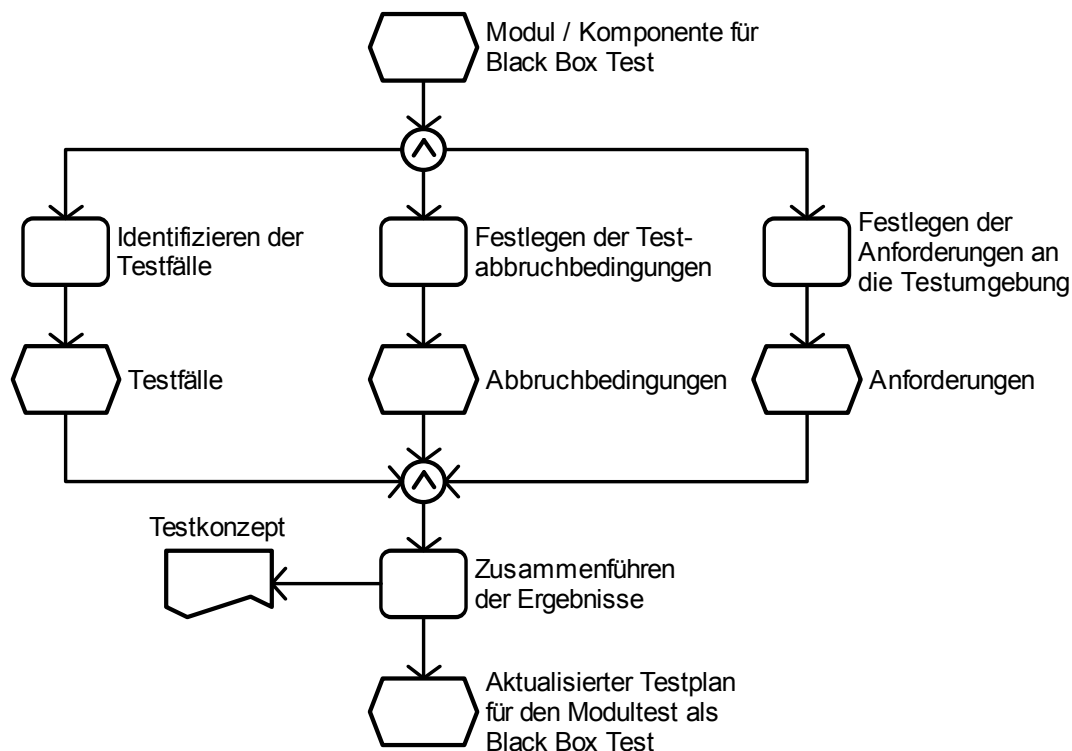


Abbildung 7: Instanzieren des Modultests (als Black Box Test).

Diese Aufgabe lässt sich in die drei Teilaufgaben „Identifizieren der Testfälle“, „Festlegen der Testabbruchbedingungen“ und „Festlegen der Anforderungen an die Testumgebung“ zerlegen. Die Aufgabe „Zusammenführen der Ergebnisse“ liefert die Testanforderungen und die Abbruchbedingungen als Ergebnis in einem Testplan.

Bei einem Black Box Test wird das zu testende Objekt nur von außen (als Black Box) betrachtet. Es werden die Testfälle gemäß der Spezifikation der Testobjekte gebildet. Als zentrale Methode zur Identifizierung der Testfälle dient das Äquivalenzklassen-Verfahren.

Testabbruchbedingungen werden definiert, um bei instabiler Software unnötige Testaufwände zu vermeiden. Testabbruchbedingungen (TAB) werden deshalb vor allem dann verwandt, wenn zeitaufwändige manuelle Testdurchführungs-, aber auch Testspezifikations- und Testimplementierungsarbeiten geplant sind. Beim Modultest wird die Testdurchführung in der Regel automatisiert durchgeführt. Die Tests laufen in der Regel automatisch beim Build mit oder werden als Batch gestartet. Der Hauptaufwand der Testdurchführung entsteht durch die Auswertung der Testergebnisse. Für die Testdurchführung werden hier deshalb eher keine TAB definiert.

3.1.4.5.1 Tätigkeiten: Instanzieren des Modultests (als Black Box Test)

- Identifizieren der Testfälle für den Modultest auf Basis der Spezifikation des Testobjekts
- Festlegen der Testabbruchbedingungen gemäß der Testplanung oder den Richtlinien aus dem QM-System
- Festlegen der Anforderungen an die Testumgebung

- Zusammenführen der Ergebnisse in einem Testkonzept als Ergebnisdokument dieses Teilprozesses

3.1.4.5.2 Kompetenzfelder: Instanzieren des Modultests (als Black Box Test)

Fähigkeiten/Fertigkeiten

- Testfälle für den Modultest im Black Box Testverfahren identifizieren können
- Testabbruchbedingungen festlegen können
- Testkonzept erstellen können
- Anforderungen an Testumgebungen festlegen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- logisch denken können
- abstrakt denken können
- Zusammenhänge erkennen können
- Kreativität im Generieren von Testfällen

Wissen

- Aufbau und Struktur von Testspezifikationen
- Testabbruchbedingungen
- Fehlerklassen
- Basiswissen zum Konfigurationsmanagement

Werkzeuge/Methoden

- QM-System
- Entwicklungswerkzeuge

3.1.4.5.3 Beispiel: Instanzieren des Modultests (als Black Box Test)

Diese Phase dient als Planungsphase für den Modultest als Black Box Test. In dem in Abschnitt 3.1.4.1 erstellten Testkonzept wurden Qualitätsziele und Teststrategie festgelegt.

Im ersten Zyklus des Modultests wurden die Festlegungen des Testkonzepts verfeinert. Zusätzlich entstand in dieser Phase ein Plan, der die Anzahl der geplanten Testzyklen und die Aufteilung der Testaktivitäten auf die einzelnen Testzyklen enthält (→ Testplan). Als Basis dafür diente eine Liste der Schnittstellen des Moduls, die im Modultest getestet werden sollte.

Entsprechend der Aufgabenstellung konzentrieren sich die Modultest-Aktivitäten auf den Test der Schnittstellen des Moduls.

3.1.4.6 Spezifizieren des Modultests (als Black Box Test)

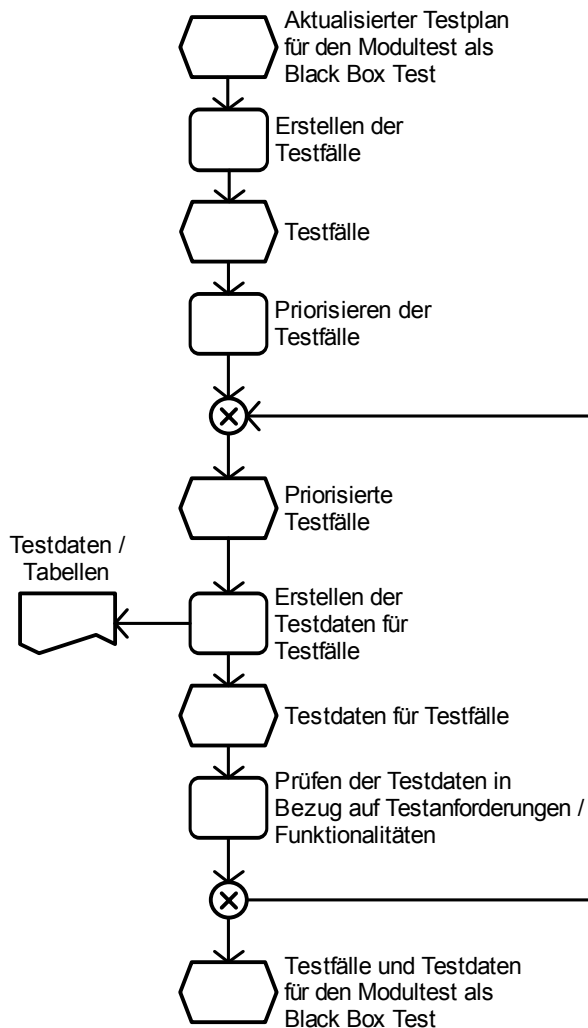


Abbildung 8: Spezifizieren des Modultests (als Black Box Test).

In diesem Teilprozess werden die Testfälle sowie die dazugehörigen Testdaten festgelegt. Anschließend werden die Testfälle priorisiert. Zum Schluss wird zum Beispiel durch Reviews überprüft, ob Testfälle und Testdaten die gewünschten Testanforderungen wie etwa die Testabdeckung erreichen.

3.1.4.6.1 Tätigkeiten: Spezifizieren des Modultests (als Black Box Test)

- Erstellen der Testfälle
- Priorisieren der Testfälle; trotz Testplanung mit Zeit und Personalplanung ist eine Priorisierung der Testfälle sinnvoll, um die wichtigsten Testfälle (Prio1) auf jeden Fall bei einer Zeitüberschreitung getestet zu haben
- Erstellen der Testdaten für Testfälle
- Prüfen der Testdaten in Bezug auf die Testanforderungen/Funktionalitäten

3.1.4.6.2 Kompetenzfelder: Spezifizieren des Modultests (als Black Box Test)

Fähigkeiten/Fertigkeiten

- Testdaten für die Testfälle erstellen können
- Testfälle erstellen können
- Testfälle priorisieren können
- Überdeckung der Testfälle mit Testanforderungen überprüfen können
- Sachverhalte in unterschiedlichen Kontexte einordnen können
- Prioritäten setzen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- kreativ denken können
- logisch denken können
- abstrakt denken können
- methodisch denken können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebung
- Software-Entwicklungsprozesse

Werkzeuge/Methoden

- Code-Überdeckungsmethoden und Werkzeuge zur Messung der Code-Überdeckung
- Code Metriken
- Testmethoden für Modultest
- Programmierumgebungen/Compiler/Interpreter
- QM-System

3.1.4.6.3 Beispiel: Spezifizieren des Modultests (als Black Box Test)

Die Testspezifikation für ein in Java implementiertes Modul im Modultest wurde im Beispielprojekt anhand der festgelegten Teststrategie erstellt. Die Testfälle wurden dabei mittels der Äquivalenzklassenmethode für die Interface-Klassen und Methoden der Schnittstelle des Moduls entwickelt. Dabei mussten die Testfälle eine funktionale Abdeckung der Schnittstellenbeschreibung des Moduls gewährleisten.

3.1.4.7 Implementieren des Modultests (als Black Box Test)

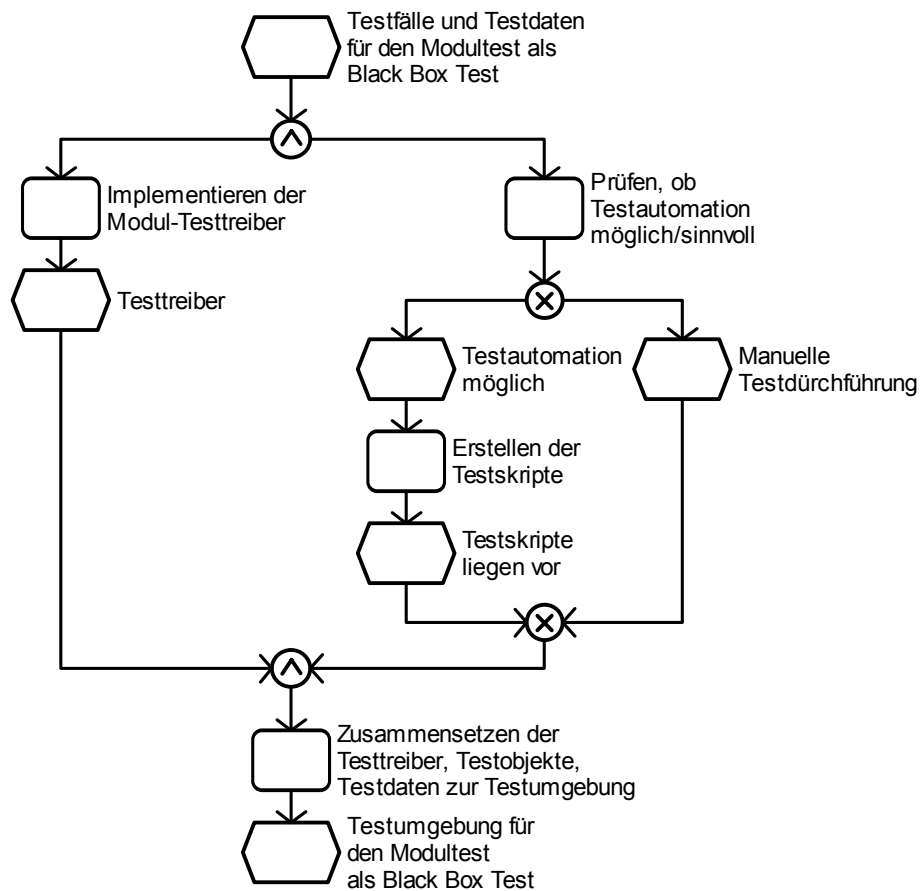


Abbildung 9: Implementieren des Modultests (als Black Box Test).

In diesem Teilprozess wird die Entwicklung der Testumgebung und die Überprüfung der Testfälle auf Testautomation beschrieben. Dabei wird die Testumgebung mithilfe der Schritte „Implementieren der Testtreiber“ und „Zusammensetzen der Testtreiber, Testobjekte, Testdaten zur Testumgebung“ erstellt. Dieser Schritt wird in enger Zusammenarbeit mit dem IT Configuration Coordinator durchgeführt, da der IT Configuration Coordinator die Testobjekte für die Testumgebung bereitstellt. Parallel hierzu wird überprüft, ob eine Testautomation für Testfälle sinnvoll ist und für die entsprechenden Testfälle werden dann die Testskripte erstellt.

3.1.4.7.1 Tätigkeiten: Implementieren des Modultests (als Black Box Test)

- Implementieren der Modultesttreiber für das Testobjekt
- Zusammensetzen der Testtreiber/Testobjekte/Testdaten zur Testumgebung
- Prüfen, ob Testautomation möglich/sinnvoll ist
- Erstellen der Testskripte; in Abhängigkeit vom eingesetzten Testwerkzeug werden Testskripte für die Testfälle erstellt

3.1.4.7.2 Kompetenzfelder: Implementieren des Modultests (als Black Box Test)

Fähigkeiten/Fertigkeiten

- Testtreiber implementieren können
- Testumgebung erstellen können
- Testskripte erstellen können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können
- programmieren können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebungen
- Software-Entwicklungsprozesse

Werkzeuge/Methoden

- Code-Überdeckungsmethoden und Werkzeuge zur Messung der Code-Überdeckung
- Code Metriken
- Testmethoden für Modultests
- Programmierumgebungen/Compiler/Interpreter
- Testautomationswerkzeuge
- QM-System

3.1.4.7.3 Beispiel: Implementieren des Modultests (als Black Box Test)

Für das zu testende Java-Modul wurde mit J-Unit eine Testumgebung erstellt, mit deren Hilfe die Interface-Klassen und Methoden parametrisiert werden konnten. Eine Testautomatisierung für die Testfälle wurde im Beispielprojekt nicht vorgenommen.

3.1.4.8 Durchführen des Modultests (als Black Box Test)

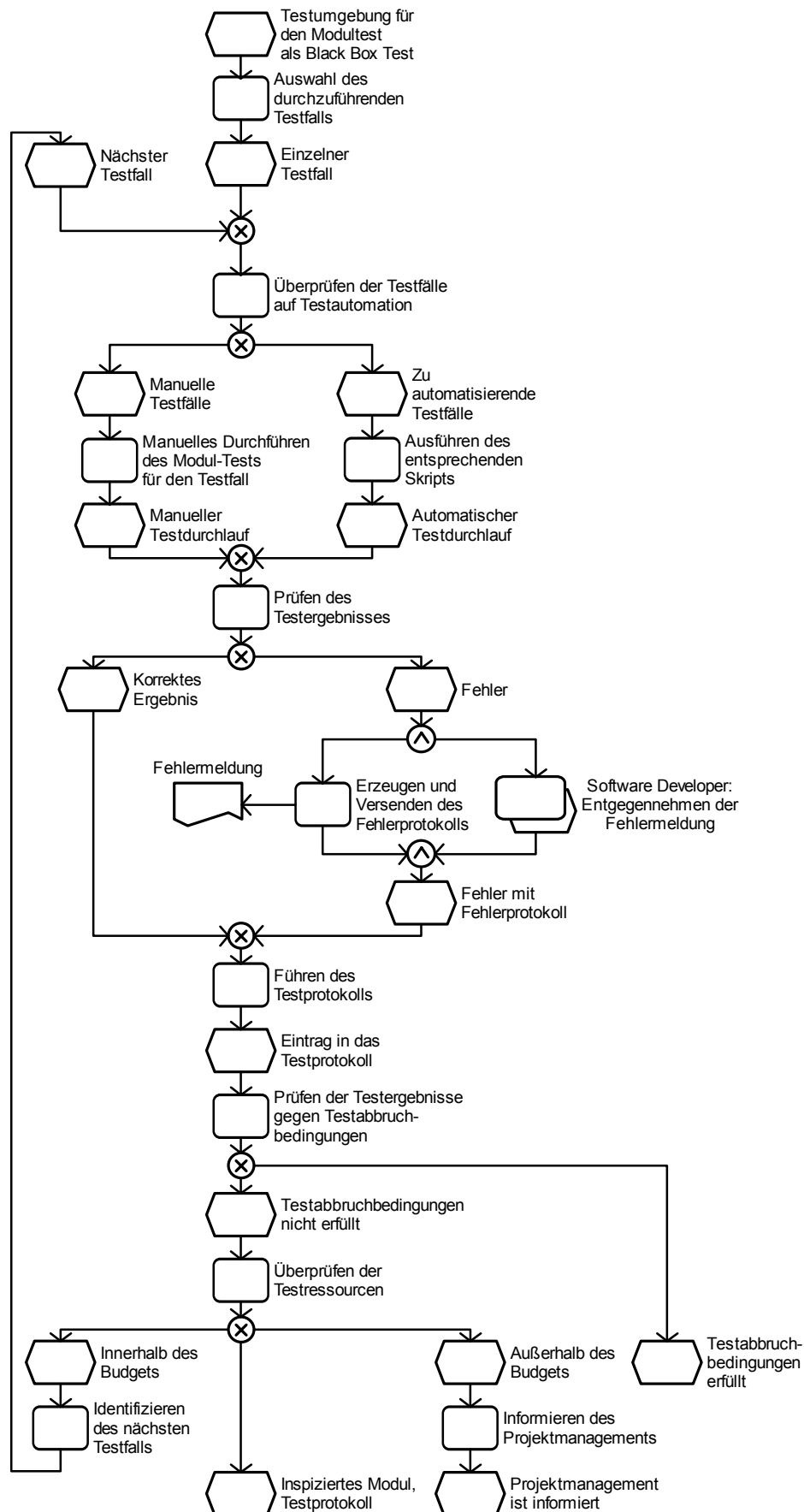


Abbildung 10 (vorherige Seite): Durchführen des Modultests (als Black Box Test).

In diesem Teilprozess existieren vier Aufgaben für jeden einzelnen Testfall. Der erste Schritt ist die Auswahl des durchzuführenden Testfalls. Danach wird für jeden Testfall unterschieden, ob er manuell oder automatisiert durch Testskripte durchgeführt wird. Anschließend muss das Ergebnis überprüft und ins Testprotokoll eingetragen werden. Im Fehlerfall muss ein Fehlerreport erstellt und an das Entwicklungsteam weitergeleitet werden. Zum Abschluss des Teilprozesses ist eine Prüfung des Testergebnisses gegen Testabbruchbedingungen durchzuführen.

3.1.4.8.1 Tätigkeiten: Durchführen des Modultests (als Black Box Test)

- Auswahl des durchzuführenden Testfalls
- Überprüfen auf Testautomation; hierbei wird nur überprüft, ob ein Testskript für die Testautomatisierung existiert, das ausgeführt werden soll
- manuelles Durchführen des Modultests für den Testfall
- Ausführen des entsprechenden Testskripts
- Prüfen des Testergebnisses auf Fehler; das Testergebnis wird gegen das spezifizierte Ergebnis geprüft
- Erzeugen und Versenden des Fehlerprotokolls; für einen erkannten Fehler wird ein Fehlerprotokoll erstellt und an das Entwicklungsteam verschickt
- Führen des Testprotokolls; die Ergebnisse jedes Testfalls werden im Testprotokoll festgehalten; dazu werden die Testdaten und die Version des Testobjekts im Testprotokoll dokumentiert
- Prüfen des Testergebnisses gegen Testabbruchbedingungen, ob die Anzahl der Fehler eine Testfortführung erlauben
- Überprüfen der Ressourcen; nach der Durchführung des Testfalls wird überprüft, ob noch genügend Ressourcen für den nächsten Testfall zur Verfügung stehen
- Identifizieren des nächsten Testfalls; anhand der Priorisierung der Testfälle wird der nächste Testfall für die Durchführung gewählt
- Informieren des Projektmanagements über nicht ausreichende Ressourcen

3.1.4.8.2 Kompetenzfelder: Durchführen des Modultests (als Black Box Test)

Fähigkeiten/Fertigkeiten

- manuell Testfälle durchführen können
- automatisiertes Testen durchführen können
- Testergebnisse gegen Testabbruchbedingungen prüfen können
- Testressourcen überwachen können
- Testfälle auswählen können
- Fehlerprotokoll erstellen können
- Testprotokoll führen können
- Reihenfolge der Testfälle anpassen können
- Testabbruch begründen und kommunizieren können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können
- analysieren können

- Genauigkeit/Sorgfalt
- Objektivität
- Belastbarkeit

Wissen

- Aufbau und Struktur von Fehlerprotokollen
- Aufbau und Struktur von Testprotokollen

Werkzeuge/Methoden

- Methoden zur Testautomation
- Werkzeug zur Erstellung und Verwaltung von Fehlerprotokollen (Trouble Ticket Systems)
- Werkzeug zur automatisierten Testdurchführung

3.1.4.8.3 Beispiel: Durchführen des Modultests (als Black Box Test)

In diesem Projekt wurde die erstellte Testautomatisierung in den regulären Build integriert. Die Testdurchführung besteht im Wesentlichen in der Auswertung der von der Testautomatisierung erstellten Testprotokolle. Der IT Test Coordinator muss entscheiden, ob es sich dabei um Fehler in der Testspezifikation, Fehler in der Testautomatisierung bzw. der Testumgebung oder aber um einen gefundenen Bug im zu testenden Modul handelt. Die Verantwortung für die Testdurchführung lag im Beispielprojekt in der Verantwortung des Kunden.

Prinzipiell unterscheidet sich die Auswertung der automatisierten Testdurchführung nicht von der manuellen Testdurchführung, wie sie in Teilprozess 3.1.4.16 „Mitwirken beim Systemtest“ beschrieben ist.

3.1.4.9 Instanziieren des Integrationstests

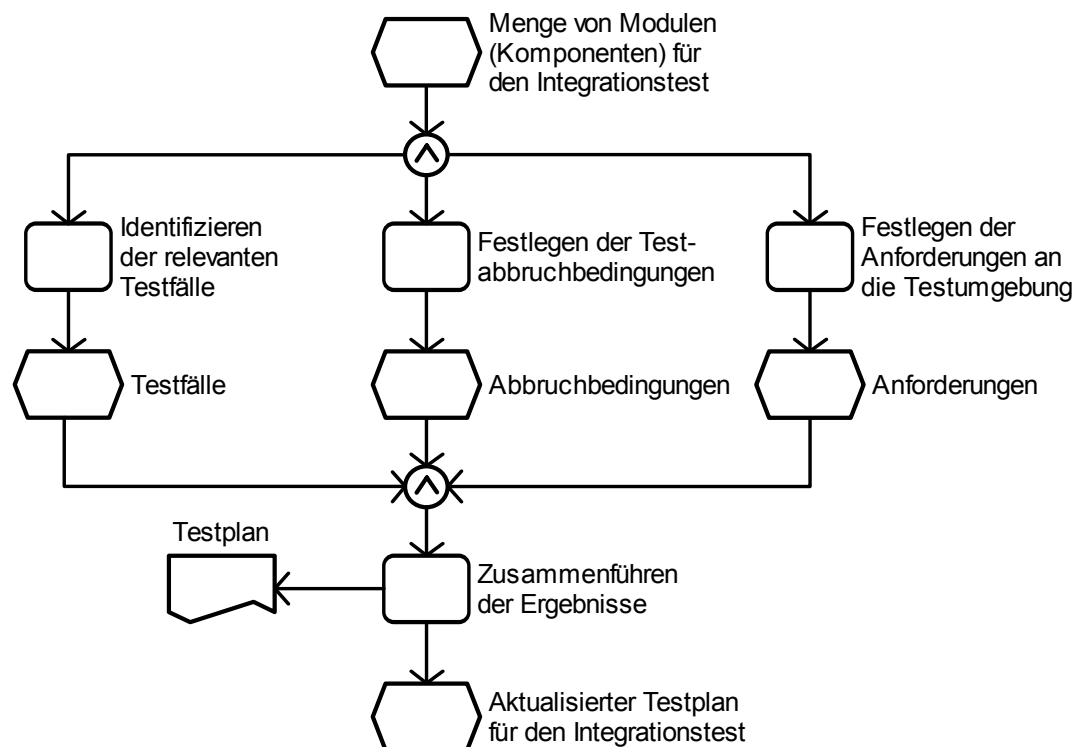


Abbildung 12: Instanziieren des Integrationstests.

Diese Aufgabe lässt sich in die Teilaufgaben „Identifizieren der Testfälle“, „Festlegen der Testabbruchbedingungen“ und „Festlegen der Anforderung an die Testumgebung“ zerlegen. Die Aufgabe „Zusammenführen der Ergebnisse“ liefert dann als Ergebnis dieses Teilprozesses den aktualisierten Testplan für den Integrationstest. Über die Testfälle können weiterhin die Testabbruchbedingungen festgelegt werden, entweder über die Anzahl der Fehler in den Fehlerklassen oder über die Aussagen, welche Testfälle auf alle Fälle fehlerfrei funktionieren müssen.

Der Schwerpunkt beim Integrationstest liegt in der Überprüfung der Schnittstellen der Module und dem Zusammenspiel der von den einzelnen Modulen bereitgestellten Funktionalitäten. Vor diesem Hintergrund sind die Testfälle und Testabbruchbedingungen für einen Integrationstest zu wählen.

3.1.4.9.1 Tätigkeiten: Instanziieren des Integrationstests

- Identifizieren der relevanten Testfälle für das Testobjekt; eine Möglichkeit für die Identifizierung der Testfälle sind die Use Cases aus der Anforderungsdefinition
- Festlegen der Testabbruchbedingungen für den Integrationstest; die Testabbruchbedingungen werden auf der Basis des Testplans und der QM-Richtlinien für den Integrationstest festgelegt und im Testkonzept dokumentiert
- Festlegen der Anforderungen an die Testumgebung; über die Anforderungen legt der IT Configuration Coordinator die Version des Testobjektes fest, das in die Testumgebung übernommen wird
- Zusammenführen der Ergebnisse; die einzelnen Ergebnisse dieses Teilprozesses werden in einer Testkonzeption dokumentiert

3.1.4.9.2 **Kompetenzfelder: Instanzieren des Integrationstests**

Fähigkeiten/Fertigkeiten

- Testfälle für den Integrationstest identifizieren können
- Anforderungen für die Testumgebung definieren können
- Testabbruchbedingungen festlegen können
- Testkonzeption erstellen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- logisch denken können
- abstrakt denken können
- Zusammenhänge erkennen können
- Kreativität im Generieren von Testfällen

Wissen

- Vorgaben für Testabbruchbedingungen beim Integrationstest
- Aufbau und Struktur von Testspezifikationen
- Fehlerklassen
- aus RUP: Methoden zur Use-Cases-Analyse [RUP]
- Basiswissen zum Konfigurationsmanagement

Werkzeuge/Methoden

- QM-System

3.1.4.9.3 **Beispiel: Instanzieren des Integrationstests**

In dieser Phase wurden die Qualitätsziele und Teststrategien für den Integrationstest des Beispielprojekts gemäß dem in Abschnitt 3.1.4.1 erstellten Testkonzept festgelegt.

Im ersten Zyklus des Integrationstests werden die Festlegungen des Testkonzepts für den Integrationstest verfeinert. Zusätzlich entsteht in dieser Phase ein Plan, der die Anzahl der geplanten Testzyklen und die Aufteilung der Testaktivitäten auf die einzelnen Testzyklen enthält (→ Testplan). Als Basis dafür dient Use Cases für die Komponenten. Die Aufstellung enthält bereits eine Planung der Testaufwände. Sie wird als Basis zur Ressourcenplanung verwendet.

Im ersten Testzyklus wurde eine initiale Planung erstellt. Während der Folgezyklen wird geprüft, ob der Plan eingehalten wurde.

Ein Testplan mit den verschiedenen Testzyklen wird im Teilprozess 3.1.4.13 „Instanzieren des Systemtests“ beschrieben.

Literatur:

[RUP] Anwendung des „Rational Unified Process (RUP)",
<http://www.rational.com/products/rup/index.jsp>, 2002.

3.1.4.10 Spezifizieren des Integrationstests

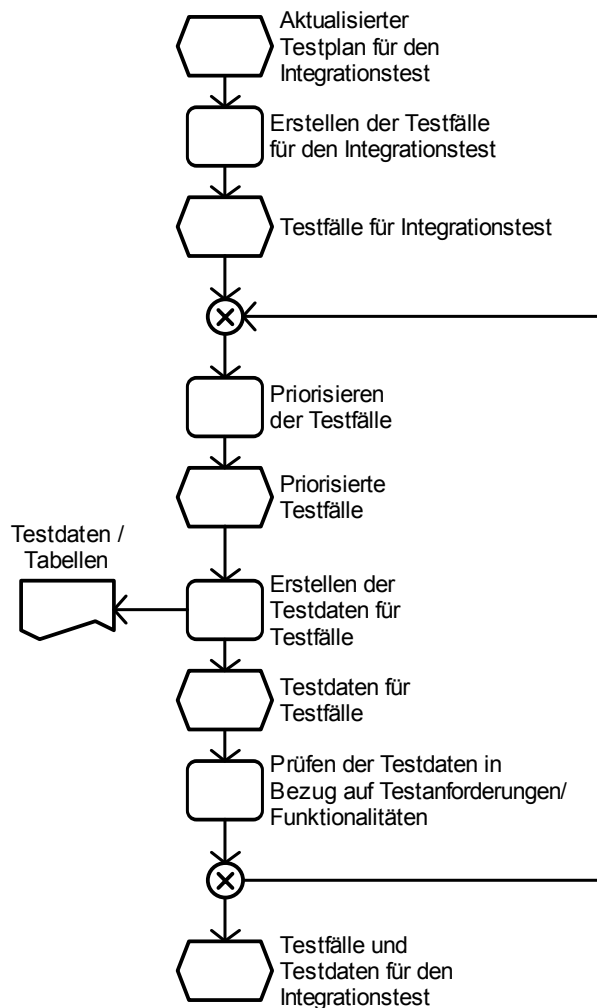


Abbildung 13: Spezifizieren des Integrationstests.

In diesem Teilprozess werden die Testfälle sowie die dazugehörigen Testdaten festgelegt. Anschließend werden die Testfälle priorisiert. Zum Schluss wird zum Beispiel durch Reviews überprüft, ob Testfälle und Testdaten die gewünschten Testanforderungen wie etwa die Testabdeckung erreichen.

Im Integrationstest wird das Zusammenspiel mehrerer Module getestet. Dabei wird das Verhalten der Module an den Schnittstellen betrachtet. Da sich dieses nur bei der Auswahl der Testfälle ausgewirkt hat, ist der Ablauf in diesem Teilprozess identisch mit den anderen Teilprozessen, in denen die Modul- oder Systemtests spezifiziert werden.

3.1.4.10.1 Tätigkeiten: Spezifizieren des Integrationstests

- Erstellen der Testfälle für den Integrationstest
- Priorisieren der Testfälle; die Testfälle werden priorisiert, damit bei einem Ressourcenproblem innerhalb der Teststufe die wichtigsten Integrationstests durchgeführt wurden
- Erstellen der Testdaten für Testfälle
- Prüfen der Testdaten in Bezug auf die Testanforderungen/Funktionalitäten

3.1.4.10.2 Kompetenzfelder: Spezifizieren des Integrationstests

Fähigkeiten/Fertigkeiten

- Testfälle und Testdaten erstellen können
- Priorisierung der Testfälle vornehmen können
- Überdeckung der Testfälle mit Testanforderungen überprüfen können
- Sachverhalte in verschiedene Kontexte einordnen können
- Prioritäten setzen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- kreativ denken können
- logisch denken können
- abstrakt denken können
- methodisch denken können

Wissen

- aus RUP: Methoden zur Use-Cases-Analyse

Werkzeuge/Methoden

- Testmethoden
- QM-System

3.1.4.10.3 Beispiel: Spezifizieren des Integrationstests

Die Testspezifikation wurde im Beispielprojekt anhand der festgelegten Teststrategie erstellt. Die Testfälle der Testspezifikation wurden aus den Use Cases der zu testenden Komponenten ermittelt. Im Beispielprojekt, in dem neue Komponenten in ein Produkt eingebettet wurden, wurde die Äquivalenzklassenmethode zur Parametrierung der Use Cases verwendet (siehe auch 3.1.4.13.3).

3.1.4.11 Implementieren des Integrationstests

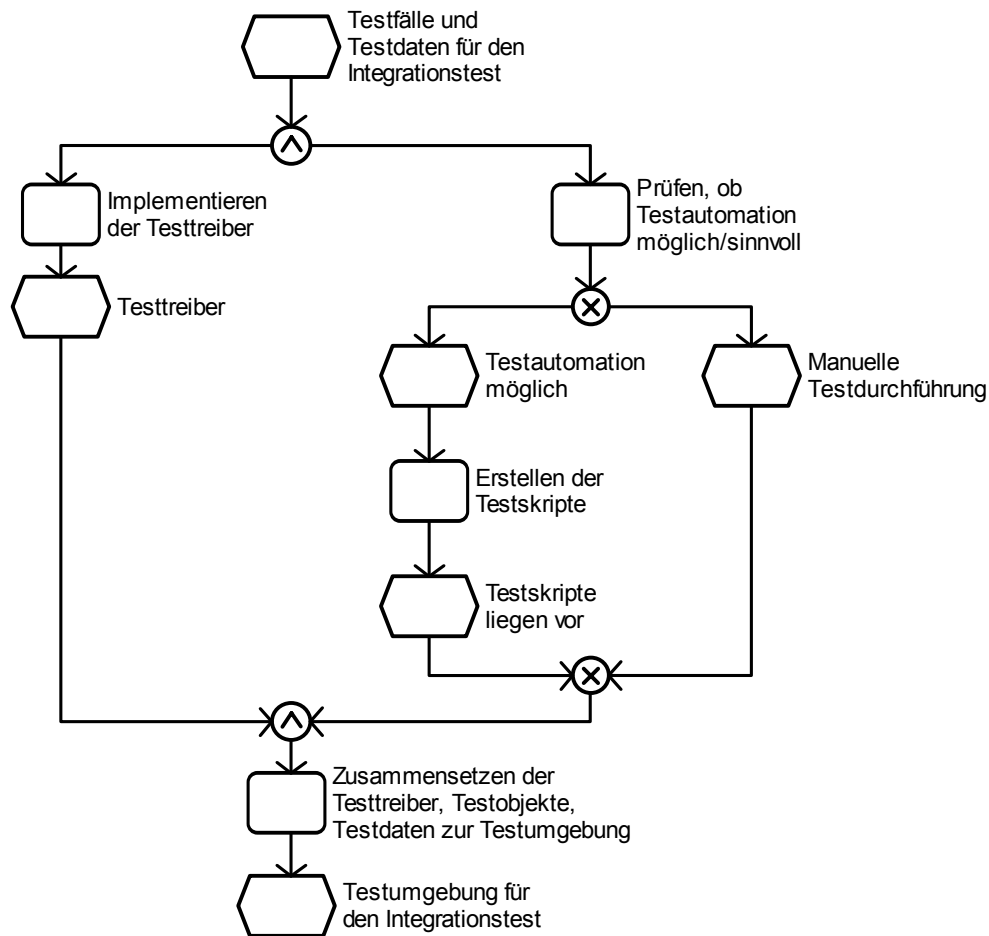


Abbildung 14: Implementieren des Integrationstests.

In diesem Teilprozess wird die Entwicklung der Testumgebung und die Überprüfung der Testfälle auf Testautomation beschrieben. Dabei wird die Testumgebung mithilfe der Schritte „Implementieren der Testtreiber“ und „Zusammensetzen der Testtreiber, Testobjekte, Testdaten zur Testumgebung“ erstellt. Dieser Schritt wird in enger Zusammenarbeit mit dem IT Configuration Coordinator durchgeführt, da dieser die Testobjekte für die Testumgebung bereitstellt. Parallel hierzu wird überprüft, ob eine Testautomation für Testfälle sinnvoll ist und für die entsprechenden Testfälle werden dann die Testskripte erstellt.

3.1.4.11.1 Tätigkeiten: Implementieren des Integrationstests

- Implementieren der Testtreiber für den Integrationstest; um einzelne Testfälle im Integrationstest durchführen zu können, werden Testtreiber benötigt, da es sich bei dem Test um kein komplettes System, sondern nur um ein Teilsystem handelt
- Zusammensetzen der Testtreiber/Testobjekte/Testdaten zur Testumgebung
- Prüfen, ob Testautomation möglich/sinnvoll ist; jeder Testfall wird auf eine Testautomation durch Abschätzen des Aufwands für das Automationsskript gegenüber dem Aufwand des manuellen Testens; relevante Faktoren sind die Anzahl der Testwiederholungen, Automationsaufwand, Ressourcen
- Erstellen der Testskripte; in Abhängigkeit vom eingesetzten Testwerkzeug werden Testskripte für die Testfälle erstellt

3.1.4.11.2 Kompetenzfelder: Implementieren des Integrationstests

Fähigkeiten/Fertigkeiten

- Testtreiber implementieren können
- Testumgebung aufsetzen können
- Aufwandsschätzung für die Testautomation vornehmen können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebungen
- Testautomatisierung

Werkzeuge/Methoden

- Testautomatisierungswerkzeuge
- Aufwandsschätzmethoden für die Testautomation

3.1.4.11.3 Beispiel: Implementieren des Integrationstests

Das Implementieren einer Testumgebung ist in Abschnitt 3.1.3.14.3 beschrieben.

3.1.4.12 Durchführen des Integrationstests

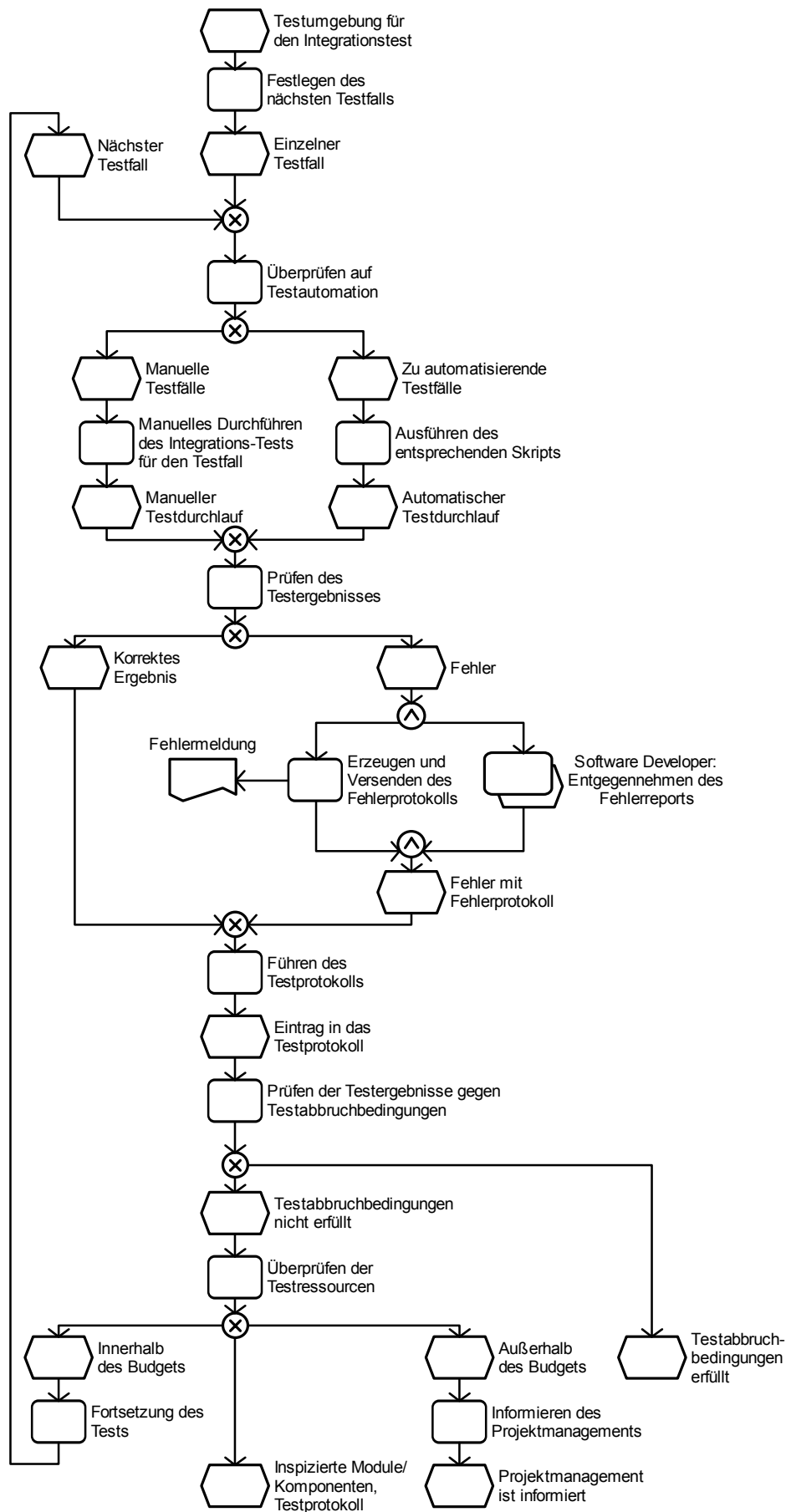


Abbildung 15 (vorherige Seite): Durchführen des Integrationstests.

In diesem Teilprozess existieren vier Aufgaben für jeden einzelnen Testfall. Im ersten Schritt muss eine Auswahl des nächsten durchzuführenden Testfalls vorgenommen werden. Danach wird für jeden Testfall unterschieden, ob der Testfall manuell oder automatisiert durch Testskripte durchgeführt wird. Anschließend muss das Ergebnis überprüft und ins Testprotokoll eingetragen werden. Im Fehlerfall muss ein Fehlerreport erstellt und an das Entwicklungsteam weitergeleitet werden. Zum Abschluss des Teilprozesses ist eine Prüfung der Testergebnisse gegen die Testabbruchbedingungen durchzuführen.

Bei der Durchführung des Integrationstests wird das Zusammenspiel mehrerer Module getestet. Dazu wird bei der Durchführung des Integrationstests das Teilsystem im Black-Box-Verfahren getestet. Der Prozessablauf ist dabei identisch mit den Teilprozessen für die Durchführung von Modul- und Systemtests.

3.1.4.12.1 Tätigkeiten: Durchführen des Integrationstests

- Festlegen des nächsten Testfalls für den Integrationstest; trotz Testplanung mit Zeit und Personalplanung ist eine Auswahl der durchzuführenden Testfälle sinnvoll, um die wichtigsten Testfälle (Prio1) auf jeden Fall bei einer Zeitüberschreitung in der Teststufe getestet zu haben
- Überprüfen auf Testautomation; hierbei wird nur überprüft, ob ein Testskript für die Testautomatisierung existiert, das ausgeführt werden soll, oder ob es sich um einen manuellen Testfall handelt
- manuelles Durchführen des Integrationstests für den Testfall
- Ausführen des entsprechenden Skripts
- Prüfen des Testergebnisses; das Testergebnis wird gegen das spezifizierte Ergebnis geprüft
- Erzeugen und Versenden des Fehlerprotokolls; für einen erkannten Fehler wird ein Fehlerprotokoll erstellt und an das Entwicklungsteam verschickt
- Führen des Testprotokolls; die Ergebnisse jedes Testfalls werden im Testprotokoll festgehalten; dazu werden die Testdaten, die Version des Testobjekts im Testprotokoll dokumentiert
- Prüfen des Testergebnisses gegen Testabbruchbedingungen; hierbei wird überprüft, ob die Anzahl der Fehler eine Testfortführung erlaubt
- Überprüfen der Ressourcen; nach der Durchführung des Testfalls wird überprüft, ob noch genügend Ressourcen für den nächsten Testfall zur Verfügung stehen
- Identifizieren des nächsten Testfalls; anhand der Priorisierung der Testfälle wird der nächste Testfall für die Durchführung gewählt
- Informieren des Projektmanagements über nicht ausreichende Ressourcen

3.1.4.12.2 Kompetenzfelder: Durchführen des Integrationstests

Fähigkeiten/Fertigkeiten

- manuell Testfälle durchführen können
- automatisiertes Testen durchführen können
- Testergebnisse gegen Testabbruchbedingungen prüfen können
- Testressourcen überwachen können
- Testfälle auswählen können
- Fehlerprotokoll erstellen können

- Testprotokoll führen können
- Reihenfolge der Testfälle anpassen können
- Testabbruch begründen und kommunizieren können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können
- analysieren können
- Genauigkeit/Sorgfalt
- Objektivität
- Belastbarkeit

Wissen

- Aufbau und Struktur von Fehlerprotokollen
- Aufbau und Struktur von Testprotokollen

Werkzeuge/Methoden

- Methoden zur Testautomation
- Werkzeug zur Erstellung und Verwaltung von Fehlerprotokollen (Trouble Ticket Systems)
- Werkzeug zur automatisierten Testdurchführung

3.1.4.12.3 Beispiel: Durchführen des Integrationstests

Die Durchführung eines Tests (Systemtests) im Beispielprojekt ist in Abschnitt 3.1.4.15.3 beschrieben.

3.1.4.13 Instanziieren des Systemtests

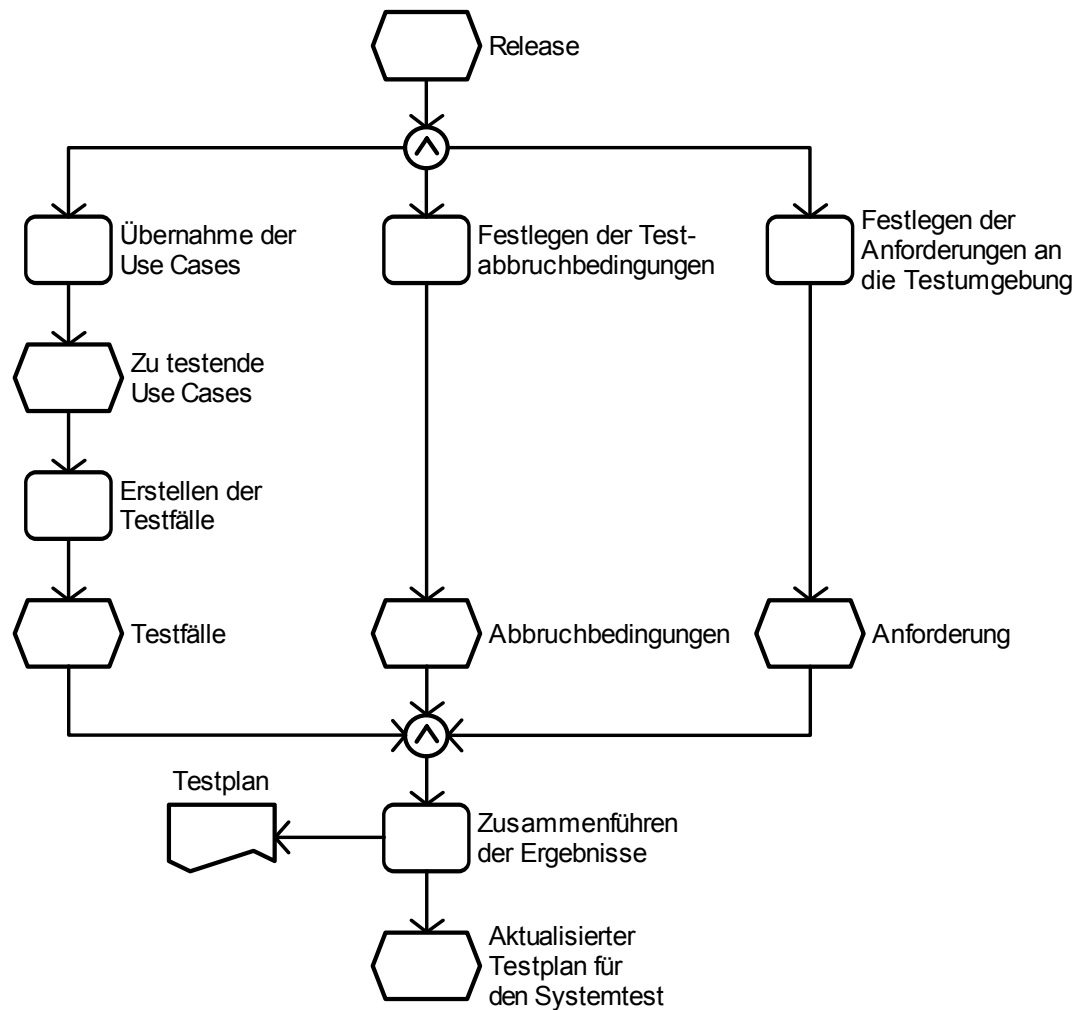


Abbildung 16: Instanziieren des Systemtests.

Die Vorbereitung des Systemtests lässt sich in die Aufgaben „Festlegung der Testabbruchbedingungen“, „Übernahme der Use Cases“, „Festlegung der Anforderungen an die Testumgebung“ und „Erstellung der Testfälle für die Use Cases“ zerlegen, wobei auf Testfälle aus Integrationstests zurückgegriffen werden kann. Die Aufgabe „Zusammenführen der Ergebnisse“ liefert als Ergebnisdokument einen Testplan mit den durchzuführenden Testfällen für den Systemtest.

Im Systemtest wird die Gesamtfunktionalität des Systems im Black-Box-Verfahren getestet. Dazu werden die Testfälle aus den Use Cases der Anforderungsdefinition in Test Cases umgesetzt. Wie für jeden anderen Test auch werden die Testabbruchbedingungen (maximale Anzahl schwerer, mittlerer und leichter Fehler) festgelegt.

Die Anforderungen an die Testumgebung bilden eine Schnittstelle zum IT Configuration Coordinator, der für die Erstellung der Testumgebung zuständig ist.

3.1.4.13.1 Tätigkeiten: Instanzieren des Systemtests

- Übernahme der Use Cases; die Basis für die zu testenden Use Cases ist die Anforderungsdefinition, in der die Use Cases für das gesamte System definiert wurden
- Erstellen der Testfälle für die Use Cases
- Festlegen der Testabbruchbedingungen für den Systemtest; die Testabbruchbedingungen werden auf der Basis des Testplans, der QM-Richtlinien für Systemtests festgelegt und im Testkonzept dokumentiert
- Festlegen der Anforderungen an die Testumgebung; über die Anforderungen legt der IT Configuration Coordinator die Version des Systems in die Testumgebung überführt wird
- Zusammenführen der Ergebnisse; die einzelnen Ergebnisse dieses Teilprozesses werden in einer Testkonzeption dokumentiert

3.1.4.13.2 Kompetenzfelder: Instanzieren des Systemtests

Fähigkeiten/Fertigkeiten

- relevante Use Cases identifizieren können
- Use Cases in Testfälle umsetzen können
- Anforderungen für die Testumgebung definieren können
- Testabbruchbedingungen für den Systemtest festlegen können
- Testkonzeption erstellen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- logisch denken können
- abstrakt denken können
- Zusammenhänge erkennen können
- Kreativität im Generieren von Testfällen

Wissen

- Vorgaben für Testabbruchbedingungen beim Integrationstest
- Aufbau und Struktur von Testspezifikationen
- Fehlerklassen
- Use Cases
- Basiswissen zum Konfigurationsmanagement

Werkzeuge/Methoden

- QM-System

3.1.4.13.3 Beispiel: Instanzieren des Systemtests

Im ersten Testzyklus

Diese Phase dient als Planungsphase für den Systemtest. In dem in Abschnitt 3.1.4.1 erstellten Testkonzept wurden Qualitätsziele und Teststrategien für den Produktrelease festgelegt. Im ersten Zyklus des Systemtests werden die Festlegungen des Testkonzepts für den Systemtest verfeinert. Zusätzlich entsteht in dieser Phase ein Plan, der die Anzahl der geplanten Testzyklen und die Aufteilung der Testaktivitäten auf die einzelnen Testzyklen enthält (→ Testplan). Als Basis dafür dient die Liste der Features, die zum Zeitpunkt der erstellten Zwischenversion bereits integriert werden sollen. Die Aufstellung enthält bereits eine Planung der Testaufwände. Sie wird als Basis zur Ressourcenplanung verwendet.

Entsprechend der Aufgabenstellung durch den Auftraggeber konzentrieren sich die Systemtest-Aktivitäten auf folgende Testbereiche:

- Test der Installationsroutinen
- Test der Migration von Daten aus Vorversionen
- Regressionstests für das Produkt

		Efforts in Engineering Days (ED)	Progress in %		Efforts in Engineering Days (ED)	Progress in %
Sample Testplan Version	Test cycle TC11			Test cycle TC2		
Planned cycle dates	7/1/2002 - 7/22/2002			7/23/2002 - 8/5/2002		
Test cycle duration	16			10		
Main emphasis	Feature 1 Complete Feature 2 First			Feature 2 Complete Feature 3 Complete		
Prepare test cycle						
Prepare test cycle		1			1	
Acceptance Tests (test cases marked with Prio 0 in test spec)						
Acceptance Tests	All test cases	10	0%	All test cases	10	0%
02 Installation and Licensing						
02.1 Installation	All test cases	5	0%	Only high priority test cases	3	0%
02.2 Deinstallation	All test cases	3	0%	Only high priority test cases	2	0%
02.3 Licensing	Only high priority test cases	1	0%	none		
03 Platform tests						
03.1 Client platforms	none			none		
03.2 Server platforms	Linux Platforms	0,5	0%	none		
04 Functional Tests						
04.1 Feature1	All test cases	5	0%	Only high priority test cases	5	0%
04.2 Feature2	Test specification	5	0%	All test cases	5	0%
04.3 Feature3	none			All test cases	5	0%
Defect retest						
Defect retest	All	2	0%	All	3	0%
Total		32,5			34	

Abbildung 17: Testplan mit Testzyklen.

Im Beispielprojekt wurde folgende Teststrategie für den Systemtest festgelegt:

- Hauptziel ist die Verifikation der Requirements.
- Die Testszenarien werden auf Basis von kundentypischen Use Cases erstellt.
- Die Testumgebung soll kundentypisch sein.
- Es wird funktionales Coverage angestrebt. (Beispielsweise sollten alle möglichen Parameter der Installationsroutinen abgedeckt sein.)

In den Folgezyklen

Im ersten Testzyklus des Systemtests wurde eine initiale Planung erstellt. Während der Folgezyklen wird geprüft, ob der Plan eingehalten wurde.

Falls nötig, wird die Testplanung (→Testplan) angepasst. Insbesondere wird die Planung in folgenden Situationen angepasst:

- Testfortschritt in vorigen Testzyklen entspricht nicht der Planung
- Änderungen in der Release-Planung durch die Entwicklung, z. B.
 - das Feature/die Komponente wurde nicht rechtzeitig fertig und kann erst im nächsten Zyklus getestet werden
 - neue Features/Komponenten werden im Gegensatz zur ursprünglichen Planung integriert
 - es gibt Qualitätsprobleme für einige Bereiche der Software (Instabilitäten, hohe Anzahl von Fehlermeldungen)

3.1.4.14 Spezifizieren des Systemtests

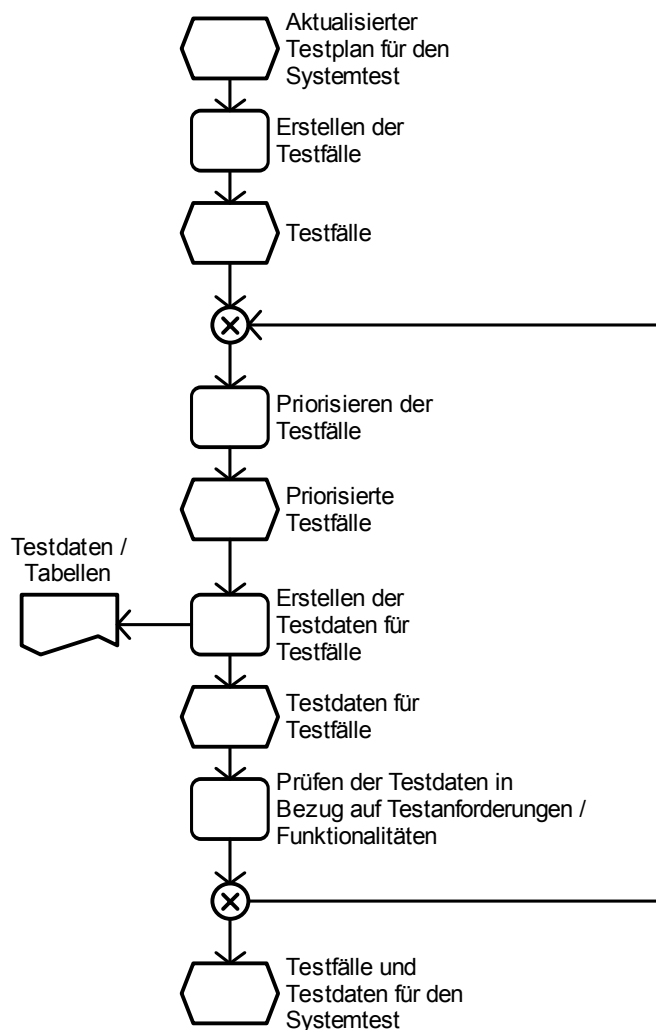


Abbildung 18: Spezifizieren des Systemtests.

In diesem Teilprozess werden die Testfälle sowie die dazugehörigen Testdaten festgelegt. Zum Schluss werden die Testdaten überprüft, ob die Testanforderungen wie z. B. Testüberdeckung durch die Testdaten abgedeckt werden.

Testfälle und Testdaten werden im Systemtest häufig durch Analyse der Anforderungsdokumentation, der funktionalen Spezifikationen sowie der Benutzerdokumentation generiert.

3.1.4.14.1 Tätigkeiten: Spezifizieren des Systemtests

Erstellen der Testfälle für den Systemtest

Priorisieren der Testfälle; die Testfälle werden priorisiert, damit bei einem Ressourcenproblem innerhalb der Testphase die wichtigsten Integrationstests durchgeführt wurden

Prüfen der Testdaten in Bezug auf die Testanforderungen/Funktionalitäten

3.1.4.14.2 Kompetenzfelder: Spezifizieren des Systemtests

Fähigkeiten/Fertigkeiten

- Testfälle und Testdaten erstellen können
- Priorisierung der Testfälle vornehmen können
- Überdeckung der Testfälle mit Testanforderungen überprüfen können
- Sachverhalte in Kontexte einordnen können
- Prioritäten setzen können
- selbstständig und zielorientiert arbeiten können
- Wesentliches von Unwesentlichem unterscheiden können
- kreativ denken können
- logisch denken können
- abstrakt denken können
- methodisch denken können
- im Team arbeiten können

Wissen

- aus RUP: Methoden zur Use-Cases-Analyse

Werkzeuge/Methoden

- Testmethoden
- QM-System

3.1.4.14.3 Beispiel: Spezifizieren des Systemtests

Die Testspezifikation wurde im Beispielprojekt anhand der in 3.1.4.13 festgelegten Teststrategie erstellt. Wie bereits beschrieben, betreut imbus die beauftragten Tests bereits über mehrere Jahre. Die Testspezifikation wurde während dieser Zeit stetig erweitert und angepasst. Zum heutigen Zeitpunkt umfasst die Spezifikation ca. 1200 Testfälle. Diese werden in einem datenbankbasierten Testmanagementsystem verwaltet.

Ein Testfall hat typischerweise folgendes Aussehen:

02.2.1.2 SAVE AS NEW NAME (new name does not exist)	
Priority:	2
Reference / Requirement:	ABC123
Configuration:	Conf 0
Precondition: empty directory dir1 File "policy0" does not exist.	
Description: Step1: // create new file in empty directory Group dir1 Select dir1 create new file using context menu "New file"; Step 2: // after editor started: change/set some field-values FILE->SAVE AS Dialog SAVE AS "policy0"; Ver. 1.0 Close editor Check: Check, that the File explorer lists the file policy0 in the directory dir1 with version 1.0. Step 3: Open file "policy0" once again in editor. SAVE AS "policy1"; Ver. 1.0 Check: Check, that the File explorer lists the files policy0 and policy1 in the directory dir1 with version 1.0.	
Postcondition: The files dir1/policy0 (Version 1.0) dir1/policy1 (Version 1.0) have been created.	

3.1.4.15 Implementieren des Systemtests

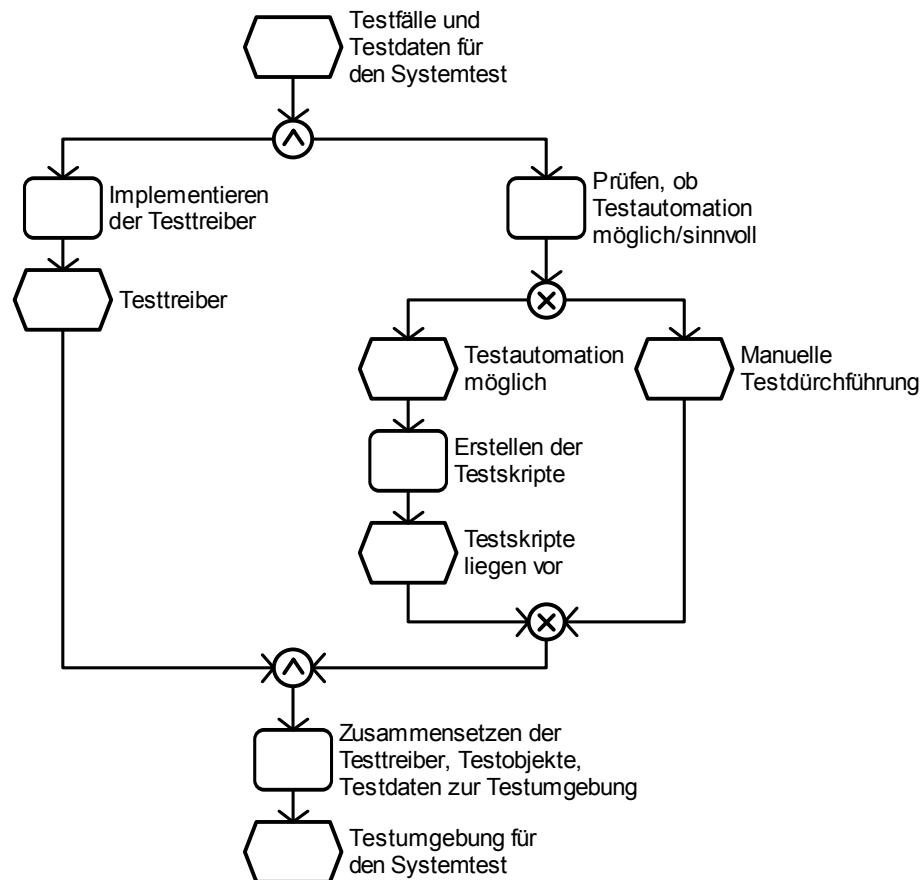


Abbildung 19: Implementieren des Systemtests.

In diesem Teilprozess wird die Entwicklung der Testumgebung beschrieben. Dabei wird die Testumgebung mithilfe der Schritte „Implementieren der Testtreiber“ und „Zusammensetzen der Testtreiber, Testobjekte, Testdaten zur Testumgebung“ erstellt. Dieser Schritt wird in enger Zusammenarbeit mit dem IT Configuration Coordinator durchgeführt, da dieser die Testobjekte für die Testumgebung bereitstellt.

Dieser Teilprozess wurde in den Referenzprozess aufgenommen, da es in einzelnen Fällen notwendig ist, für den Systemtest Testtreiber zu entwickeln. In der Regel kann der Systemtest ohne spezielle Testtreiber durchgeführt werden.

3.1.4.15.1 Tätigkeiten: Implementieren des Systemtests

- Implementieren der Testtreiber für den Systemtest; um einzelne Testfälle im Systemtest durchführen zu können, werden teilweise Testtreiber benötigt
- Zusammensetzen der Testtreiber/Testobjekte/Testdaten zur Testumgebung
- Prüfen, ob Testautomation möglich/sinnvoll ist; jeder Testfall wird auf die Existenz von Testskripten hin überprüft, die eine Testautomation des Testfalls bei der Durchführung des Systemtests ermöglichen
- Erstellen der Testskripte; in Abhängigkeit vom eingesetzten Testwerkzeug werden Testskripte für die Testfälle erstellt

3.1.4.15.2 Kompetenzfelder: Implementieren des Systemtests

Fähigkeiten/Fertigkeiten

- Testtreiber implementieren können
- Testumgebung aufsetzen können
- Aufwandsschätzung für die Testautomation vornehmen können
- selbstständig und zielorientiert arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können

Wissen

- Funktion von Testtreibern
- Programmierkenntnisse
- Aufbau und Struktur von Testumgebungen
- Testautomatisierung

Werkzeuge/Methoden

- Aufwandsschätzmethoden für die Testautomation
- Werkzeug zur automatisierten Testdurchführung

3.1.4.15.3 Beispiel: Implementieren des Systemtests

In dieser Phase wird zunächst die Testumgebung erstellt. Für den Fall der Installationstests war das Ziel, kundentypische Netzwerkkonfigurationen aufzubauen. Es wurden Verfahren zur Wiederherstellung dieser Netzwerkkonfigurationen erstellt und dokumentiert.

Des Weiteren wurden häufig durchgeführte Testfälle aus Effizienzgründen mithilfe des Testautomatisierungstools WinRunner implementiert. Das Ergebnis sind jeweils Testskripte für die zur Testautomatisierung ausgewählten Testfälle.

Die Auswahl der zu automatisierenden Testfälle basierte auf folgenden Kriterien:

- der Häufigkeit der Testdurchführung
- der Komplexität/dem Aufwand der Testautomatisierung
- der Komplexität/dem Aufwand der Testdurchführung
- der Stabilität der Schnittstelle (Abschätzung der Änderungsaufwände für die Testautomatisierung)

Abbildung 20 (vorherige Seite): Mitwirken beim Systemtest.

In diesem Teilprozess existieren vier Aufgaben für jeden einzelnen Testfall. Im ersten Schritt muss eine Auswahl des nächsten durchzuführenden Testfalls vorgenommen werden. Danach wird für jeden Testfall unterschieden, ob der Testfall manuell oder automatisiert durch Testskripte durchgeführt wird. Anschließend muss das Ergebnis überprüft und ins Testprotokoll eintragen werden. Im Fehlerfall muss ein Fehlerreport erstellt und an das Entwicklungsteam weitergeleitet werden. Zum Abschluss des Teilprozesses ist eine Prüfung der Testergebnisse gegen die Testabbruchbedingungen durchzuführen.

Der IT Test Coordinator nimmt am Systemtest teil. Die Entscheidung über die Testfortführung liegt aber beim Projektleiter. Die Aufgaben des IT Test Coordinator sind dabei die Koordinierung der Durchführung jedes Testfalls und das Protokollieren der Ergebnisse im Testprotokoll.

3.1.4.16.1 Tätigkeiten: Mitwirken beim Systemtest

- Überprüfen auf Testautomation; hierbei wird nur überprüft, ob ein Testskript für die Testautomatisierung existiert, das ausgeführt werden soll, oder ob es sich um einen manuellen Testfall handelt
- manuelles Durchführen des Systemtests für den Testfall
- Ausführen des entsprechenden Skripts
- Prüfen des Testergebnisses; das Testergebnis wird gegen das spezifizierte Ergebnis des Use Cases geprüft
- Erzeugen und Versenden des Fehlerprotokolls; für einen erkannten Fehler wird ein Fehlerprotokoll erstellt und an das Entwicklungsteam verschickt
- Prüfen des Testergebnisses gegen Testabbruchbedingungen; hierbei wird überprüft, ob die Anzahl der Fehler eine Testfortführung erlaubt
- Führen des Testprotokolls; die Ergebnisse jedes Testfalls werden im Testprotokoll festgehalten; dazu werden die Testdaten und die Version des Testobjekts im Testprotokoll dokumentiert

3.1.4.16.2 Kompetenzfelder: Mitwirken beim Systemtest

Fähigkeiten/Fertigkeiten

- manuell Testfälle durchführen können
- automatisiertes Testen durchführen können
- Testergebnisse gegen Testabbruchbedingungen prüfen können
- Testressourcen überwachen können
- Testfälle auswählen können
- Fehlerprotokoll erstellen können
- Testprotokoll führen können
- Reihenfolge der Testfälle anpassen können
- Projektleitung informieren können
- zielorientiert Arbeiten können
- Zusammenhänge erkennen können
- methodisch denken können
- analysieren können
- sich durchsetzen können

- zusammenarbeiten können
- Genauigkeit/Sorgfalt
- Teamfähigkeit
- Objektivität
- Belastbarkeit

Wissen

- Aufbau und Struktur von Fehlerprotokollen
- Aufbau und Struktur von Testprotokollen

Werkzeuge/Methoden

- Methoden zur Testautomation
- Werkzeug zur Erstellung und Verwaltung von Fehlerprotokollen (Trouble Ticket Systems)
- Werkzeug zur automatisierten Testdurchführung

3.1.4.16.3 Beispiel: Mitwirken beim Systemtest

Die Testdurchführung hat generell zwei Ergebnisse: Zum einen wird das Ergebnis der Durchführung des Testfalls dokumentiert (PASS/FAIL). Im Beispielprojekt werden die Testergebnisse für jeden Testfall im Testmanagementsystem dokumentiert.

In diesem Projekt wurde ein Teil der Testfälle automatisiert durch Testskripte durchgeführt. Das Testautomatisierungstool erstellt mit jedem Testdurchlauf einen Testreport. Aus diesem ist pro Testskript das Ergebnis des Testlaufs ersichtlich. Der Tester wertet diesen Testreport aus.

Beobachtet der Tester bei manueller oder automatisierter Testdurchführung ein Fehlverhalten in der Software, so muss er nun entscheiden, ob es sich um einen Fehler handelt. Zum Teil handelt es sich um Fehler in der Testspezifikation/Testautomatisierung oder in der Testumgebung. In diesem Fall koordiniert der IT Test Coordinator die Wartungsarbeiten.

Für Softwarefehler wurde im Beispielprojekt eine Fehlermeldung in der mit dem Kunden vereinbarten Fehlerdatenbank vorgenommen:

<u>FehlerNr</u>	imb-0002
<u>Titel</u>	Fehlerliste wird nicht aktualisiert
<u>Entdecker</u>	Jochen
<u>Reproduzierbar</u>	Ja
<u>Testfall</u>	1.2.3 Anmelden am System
<u>Erf.Dat.</u>	27.08.2002
<u>Schweregrad</u>	Medium
<u>Version</u>	7.00 / IC1
<u>Konfiguration</u>	Conf 0
<u>Maske/Zustand</u>	
<i>Detailbearbeitung</i>	
<u>Aktion</u>	
<i>neuen Status anlegen</i>	
<u>Beschreibung</u>	
<i>In der Statusliste wird der Status nicht aktualisiert.</i>	
<i>Es ist ein Requery nötig (früher hat Refresh genügt).</i>	
 <i>Nachtrag: Genau die "berechneten" Felder Priorität + Status (Zahl + Bezeichnung) werden nicht aktualisiert.</i>	
<i>Lösung: View v Statusliste macht diese "Berechnung" bereits auf dem Sql-Server, dann klappt hier die Aktualisierung.</i>	
Status	Änderung am, durch
DONE	02.09.200 Jochen
TEST	30.08.200 Andre
STDY	28.08.200 Andre
NEW	28.08.200 Jochen
Version	Bemerkung
7.01	ok
7.01	mit View gelöst
7.00	Seltsam: andere Felder werden bei .Refresh sofort übernommen.
7.00	

3.1.4.17 Kategorisieren der Fehler/Erstellen des Testberichts

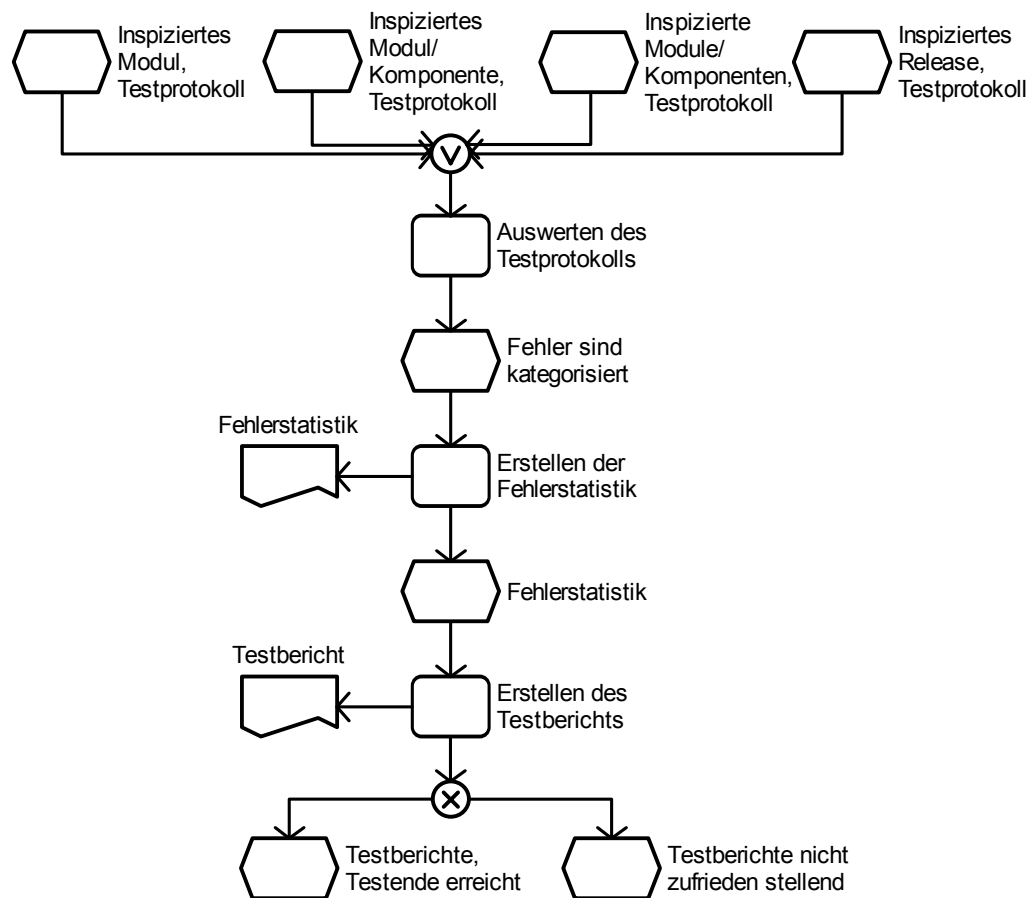


Abbildung 21: Kategorisieren der Fehler/Erstellen des Testberichts.

Dieser Teilprozess bildet den Abschluss einer jeden Teststufe (Modul-, Integrations- System-test). Er besteht im Wesentlichen aus den zwei Tätigkeiten „Kategorisierung der Fehler“ und „Erstellung des Testberichts“. Der Testbericht ist das Abschlussdokument jeder Teststufe. Im Testbericht werden der Testverlauf und die Besonderheiten des Tests beschrieben. Des Weiteren enthält der Testbericht eine Fehlerstatistik und eine Bewertung des Testobjekts. Darüber hinaus werden die Fehler der zu testenden Einheit im Testbericht dokumentiert.

Die Fehlerstatistik ist eine detaillierte Auflistung aller Testfälle mit Testergebnissen. Dabei wird jeder Fehler gemäß einer Skala, welche die Auswirkung des Fehlers auf den Testfortgang festlegt, bewertet.

3.1.4.17.1 Tätigkeiten: Kategorisieren der Fehler/Erstellen des Testberichts

- Auswerten des Testprotokolls; um einen Testbericht zu schreiben, müssen die Testergebnisse, die im Testprotokoll festgehalten wurden, ausgewertet werden
- Erstellen der Fehlerstatistik; auf der Basis des ausgewerteten Testprotokolls wird eine Fehlerstatistik erstellt, die verschiedene Sichten auf das Testprotokoll visualisiert
- Erstellen des Testberichts; der Testbericht fasst die Testergebnisse zusammen und er enthält den Status des Testobjekts

3.1.4.17.2 Kompetenzfelder: Kategorisieren der Fehler/Erstellen des Testberichts

Fähigkeiten/Fertigkeiten

- Testprotokoll auswerten können
- Fehlerstatistik erstellen können
- Testbericht erstellen können
- Testbericht präsentieren können
- Projektleitung über Testausgang informieren können
- selbstständig und zielorientiert arbeiten können
- Ergebnisse präsentieren können
- Zusammenhänge erkennen können
- Genauigkeit/Sorgfalt
- Objektivität
- Belastbarkeit

Wissen

- Aufbau und Struktur von Fehlerstatistiken
- Aufbau und Struktur von Testberichten

Werkzeuge/Methoden

- Werkzeuge zur Erstellung von Statistiken

3.1.4.17.3 Beispiel: Kategorisieren der Fehler/Erstellen des Testberichts

Als Abschlussdokument wurde in dem Projekt ein Testprotokoll mit den Testergebnissen jedes einzelnen Testfalls jedes Testzyklus' erstellt. Neben dieser detaillierten Darstellung wurde eine Zusammenfassung der Testergebnisse im Testbericht erstellt. Dabei wurden die Anzahl der Fehler, der Testaufwand und die Testunterbrechungen auf die einzelnen Komponenten aufgeschlüsselt. Neben dieser textuellen Beschreibung wurden die entsprechenden Kennzahlen in ein Excel Sheet übertragen, um so eine Auswertung und grafische Darstellung zu ermöglichen.

Diese drei Dokumente wurden nach Testabschluss dem Kunden übergeben. Damit war das Kundenprojekt abgeschlossen.

4 Glossar

Begriff	Definition
Fehlermeldung	Klare, präzise, schriftliche Beschreibung eines Fehlers oder Problems zum Zwecke der Dokumentation und Kommunikation. Idealerweise über eine Fehlerdatenbank.
Integrationstest	Test mit dem Ziel, Fehler in Schnittstellen und im Zusammenspiel zwischen integrierten Komponenten zu finden.
Komponente	<ol style="list-style-type: none"> 1. Kleinste Software-Einheit, für die eine separate Spezifikation verfügbar ist (Hinweis: Komponente wird unterschiedlich definiert, diese Definition entspricht [BS7925-1]). 2. Software-Einheit, die die Implementierungsstandards eines Komponentenmodells (EJB, CORBA, .NET) erfüllt.
Release/Konfiguration/Build	<ol style="list-style-type: none"> 1. Menge von Software-Objekten einer bestimmten Version, die zusammen ein (Teil-)System bilden. 2. Zustand der Umgebung eines Testobjekts, der als Vorbedingung für die Durchführung von Testfällen erreicht sein muss.
SystemTest	Test eines integrierten Systems um sicherzustellen, dass es spezifizierte Anforderungen erfüllt.
Testautomatisierung	<ol style="list-style-type: none"> 1. Einsatz von Software-Werkzeugen zur Erstellung bzw. Programmierung von Testfällen mit dem Ziel, die Testfälle rechnergestützt wiederholt ausführen zu können. 2. Unterstützung aller Arbeiten im Testprozess durch entsprechende Software-Werkzeuge.
Testdurchführung	Ausführung der Testfälle bzw. Testszenarien (Aktivität im Testprozess).
Testergebnis	<ol style="list-style-type: none"> 1. Alle Dokumente, die im Rahmen eines Testlaufs erstellt werden (hauptsächlich das Testprotokoll und dessen Auswertung). 2. Freigabe oder Zurückweisung des Testobjekts (je nach Anzahl und Schwere der ermittelten Fehlerwirkungen).
Testkonzept	Dokument, das den Umfang, die Vorgehensweise, die Ressourcen und die Zeitplanung der intendierten Tests (inkl. aller Aktivitäten) beschreibt (Inhalt z. B. gem. [IEEE 829]).
Testplan	<ol style="list-style-type: none"> 1. Zeitliche Planung der Testdurchführung (Zuordnung der Testfälle zu Testern und Festlegung des Durchführungszeitpunktes). 2. Verzeichnis aller Testfälle, in der Regel thematisch bzw. nach Testzielen gruppiert.
Testplanung	Aktivität im Testprozess zur Erstellung und Fortschreibung von Testkonzept und Testplan.

Testprotokoll	Schriftlich festgehaltenes Ergebnis eines Testlaufs oder einer Testsequenz (im Fall von automatisierten Tests meist vom Werkzeug erstellt). Aus dem Protokoll muss hervorgehen, welche Teile wann, von wem, wie intensiv und mit welchem Ergebnis getestet wurden.
Testspezifikation	1. Begründung der Auswahl der (logischen) Testfälle sowie die Beschreibung der (logischen) Testfälle in einem Dokument. 2. Aktivität im Testprozess.
Testumgebung	Gesamtheit aller Hard- und Software-Komponenten (auch der Testrahmen), die notwendig sind, um Testfälle durchzuführen.
Unit-/Komponenten-Test	Test einer Software-Einheit (Unit, Modul, Komponente).

Literatur:

[Spillner/Linz] Basiswissen Softwaretest : Aus- und Weiterbildung zum Certified-Tester, Andreas Spillner, Tilo Linz, Heidelberg, dpunkt-Verl.

[BS7925-1]: Britischer Standard für software testing vocabulary. Computer Society Specialist Interest Group in Software Testing (BCS SIGIST).

[IEEE 829] ANSI/IEEE Std 829-1983, IEEE Standard for Software Test Documentation, Institute of Electrical and Electronics Engineers, Inc., August 1983.