Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Thema:

**Bereitstellung und Test der Software „Progress Monitor“ zum Monitoring und zur Analyse von Dateikonvertierungen für den elektronischen Datenaustausch (EDI)**

Praxisprojekt

Fachbereich I – Wirtschafts‑ und Gesellschaftswissenschaften

Studiengang Wirtschaftsinformatik Online

Betreuender Dozent: Prof. Dr. Ing. Alexander Huber

Betreuer im Unternehmen: Antje Galinsky

vorgelegt von: Michal Wischniewski  
Matrikelnummer: 917983  
Kemnitzer Chaussee 71  
14542 Werder  
+49 151 22784419  
E‑Mail: s76826@bht‑berlin.de

Abgabetermin: 25. August 2023

Sperrvermerk

Diese Arbeit enthält vertrauliche Daten der Softzoll GmbH & Co. KG. Eine Weitergabe der Arbeit im Ganzen oder in Teilen sowie das Anfertigen von Kopien (auch digital) ‑ sind grundsätzlich untersagt. Ausnahmen bedürfen der schriftlichen Genehmigung.

Softzoll GmbH & Co. KG

Abteilung: Software Entwicklung and Integration

Kurfürstenstr. 112

10787 Berlin

Telefon: +49 (0)30 210023‑50

E-Mail: info@softzoll.de

Website: https://www.softzoll.de

Berlin, den 25. August 2023

Inhaltsverzeichnis

Inhaltsverzeichnis III

Verzeichnis der Abkürzungen und Akronyme IV

Abbildungsverzeichnis VI

Tabellenverzeichnis VII

Definition der Begriffe VIII

1 Einleitung 1

2 Methodik 2

3 Unternehmensprofil der „Softzoll GmbH & Co. KG“ 4

4 Projektbegründung 5

4.1 Projektziel und Projektumfang 5

5 Anforderungsanalyse (Requirements Engineering) 7

5.1 Risikoanalyse (Projekt Risikomanagement) 9

6 Ist-Aufnahme im Unternehmen Softzoll GmbH & Co. KG 11

7 Entwurf / Spezifikation 18

8 Implementierung 22

9 Test 23

9.1 Einführung 23

9.2 Testvorbereitung 27

9.3 Testausführung 31

9.4 Testauswertung 35

10 Fazit 37

A Anforderungen aus dem Anforderungsdokument ReqSpec 1.001 vom 26.06.2023 38

B Risikokatalog 41

C Testbericht zum Test der Software „Progress-Monitor“ 44

Literaturverzeichnis 49

Verzeichnis der Abkürzungen und Akronyme

API Application Programming Interface

AS2 Applicability Statement 2

bzw. beziehungsweise

BPMN Business Process Model and Notation

CSV Comma Separated Value

CI Continuous Integration

CD Continuous Deployment

CM Change Management

Cascading Style CSS

Sheets

DBMS Datenbankmanagementsystem

DevOps Development- und IT Operations

ERP Enterprise-Resource-Planning

GUI Graphical User Interface (Benutzeroberfläche)

EDI Electronic Data Interchange

EDIFACT Electronic Data Interchange for Administration, Commerce and Transport

ETL Extract‑Transform‑Load

Ewk Eintrittswahrscheinlichkeit

FF Fix Format

FTP File Transfer Protocol

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

Infor

IDE Integrated Development Environment   
(integrierte Entwicklungsumgebung)

IDoc Intermediate Document

JDBC Java Database Connectivity

JDE JD Edwards

JDK Java Development Kit

JRE Java Runtime Environment

KVP kontinuierlicher Verbesserungsprozess

OFTP Odette File Transfer Protocol

RC Release Candidates

RDBMS Relational Database Management System   
(relationales Datenbankverwaltungssystem)

RE Requirements Engineering

RM Requirements Management

PDF Portable Document Format

SAP Systeme Anwendungen und Produkt

SFTP Secure File Transfer Protocol

SMTP Simple Mail Transfer Protocol

SQL Structured Query Language

SVN Subversion

TRC Tradacoms

WWW World Wide Web

X12 ANSI X12

X.400 E-Mail-System basierend auf P7-Protokoll

XML Extensible Markup Language

Abbildungsverzeichnis

[**Abb. 5.1:** User Story Map 7](#_Toc141640910)

[**Abb. 6.1:** BPMN-Diagramm zum Prozess "Datenkonvertierung" 14](#_Toc141640911)

[**Abb. 9.1:** Grundlegende Teststrategie 23](#_Toc141640912)

[**Abb. 9.2:** Die vier Prüfebenen des Software-Tests 25](#_Toc141640913)

[**Abb. 9.3:** Schematischer Testablauf 27](#_Toc141640914)

[**Abb. 9.4:** Einrichten der Umgebungsvariablen im Betriebssystem des Web-Servers 29](#_Toc141640915)

[**Abb. 9.5:** Progress Monitor: Mandantenanalage im "Progress Monitor" 29](#_Toc141640916)

[**Abb. 9.6:** Progress Monitor: Benutzeranalage im "Progress Monitor" 30](#_Toc141640917)

[**Abb. 9.7:** Progress Monitor: Anlage eines Testkunden 30](#_Toc141640918)

[**Abb. 9.8:** Progress Monitor: Anlage eines Testlieferanten 30](#_Toc141640919)

[**Abb. 9.9**: Progress Monitor: Anlage eines Workflows 31](#_Toc141640920)

[**Abb. 9.10:** Schematische Darstellung eines Black-Box-Tests 32](#_Toc141640921)

[**Abb. 9.11:** Abbildung zur Testsequenz TS-6.1 aus der Testvorschrift 33](#_Toc141640922)

[**Abb. 9.12:** Übergabe der Test-Datei an den Datenkonverter, Testfall "TF-6.1.3" 34](#_Toc141640923)

[**Abb. 9.13:** Bildschirmausgabe zum Testfall 6.1.3 35](#_Toc141640924)

[**Abb. 10.1:** Testbericht zum Test der Software „Progress Monitor“, Seite 2 44](#_Toc141640925)

[**Abb. 10.2:** Testbericht zum Test der Software „Progress Monitor“, Seite 3 45](#_Toc141640926)

[**Abb. 10.3:** Testbericht zum Test der Software „Progress Monitor“, Seite 4 46](#_Toc141640927)

[**Abb. 10.4:** Testbericht zum Test der Software „Progress Monitor“, Seite 5 47](#_Toc141640928)

[**Abb. 10.5:** Testbericht zum Test der Software „Progress Monitor“, Seite 6 48](#_Toc141640929)

[**Abb. 10.6:** BPMN zum Prozess „Softwaretest“ 49](#_Toc141640930)

Tabellenverzeichnis

[**Tab**. 6.1: Übersicht zu unterstützten Datenformaten und Kommunikationsprotokollen 13](#_Toc141634014)

[**Tab. 6.2:** Übersicht zu den existierenden Systemkomponenten 15](#_Toc141634015)

[**Tab. 7.1:** Spezifikation und Entwurfsentscheidungen 18](#_Toc141634016)

[**Tab. 9.1:** Übersicht über die zu installierenden Programme 28](#_Toc141634017)

[**Tab. 10.1:** Anforderungsdokument ReqSpec 1.001 vom 26.06.2023 38](#_Toc141634018)

[**Tab. 10.2:** Risikokatalog 41](#_Toc141634019)

# Definition der Begriffe

Für das Verständnis der Arbeit und die richtige Einordnung der Bedeutung der Fachbegriffe werden diese folgend definiert.

*Apache Tomcat*

Apache Tomcat® ist eine Open Source Software und stellt die Dienste eines WWW-Servers (Web-Servers) zur Verfügung (Vgl. Alpar et al., (2023), S. 537).

*Application Programming Interface (API)*

Über ein API werden andere Programme an ein Softwaresystem angebunden, um die Funktionen dieses Systems den Programmen zur Verfügung zu stellen (Vgl. Kaufmann & Mülder, (2023), S. 241). Das API fungiert somit als Schnittstelle, die alle relevanten Informationen über das Zusammenwirken des Systems mit seiner Umgebung beschreibt (Vgl. Broy, (2023), S. 180).

*Business Process Management Notation (BPMN)*

BPMN ist eine Notation zur Modellierung von Geschäftsprozessen und dient der Analyse und Dokumentation dieser. Über grafische Symbole lassen sich komplexe Geschäftsprozesse übersichtlich darstellen (Vgl. Kaufmann & Mülder, (2023), S. 316, 322).

*Change Management (CM)*

Im CM werden nach Eintreten einer Veränderung in den Geschäftsprozessen oder in den Anforderungen alle Maßnahmen, die für die Berücksichtigung der Veränderung notwendig sind, kontrolliert eingeleitet sowie schnell und effektiv durchgeführt (Vgl. Scheer et al., (2003), S. 5).

*Continuous Delivery (CD)*

Unter CD wird in der Softwareentwicklung die kontinuierliche automatisierte Software-Auslieferung mittels der „Delivery Pipeline“ verstanden (Vgl. Alt et al., (2017), S. 27).

*Continuous Integration (CI)*

CI verfolgt das Konzept, Änderungen am Quellcode bzw. Entwicklungsfortschritte von einzelnen Entwicklern eines Teams schnell zusammenzuführen und zu testen. Dadurch werden Fehler frühzeitig erkannt und Softwareprototypen (funktionsfähige Software) in kurzer Zeit entwickelt (Vgl. Alt et al., (2017), S. 29).

*Development“ und „(IT)-Operations (DevOps)*

Der Begriff „DevOps“ setzt sich aus den Worten „Development“ und „(IT)-Operations“ zusammen und betont dabei die Zusammenarbeit dieser beiden Bereiche. Dabei werden die Geschäftsprozesse eingeschlossen und der Kundennutzen in den Mittelpunkt gestellt (Vgl. Halstenberg et al., (2020), S. 1). „Development“ und „Operations“ sind eigenständige Organisationseinheiten im IT-Bereich mit unterschiedliche Zielsetzungen (Vgl. Alt et al., (2017), S. 23). Beim DevOps-Ansatz wird auf die ganzheitliche Zusammenarbeit zwischen der Softwareentwicklung, dem IT-Betrieb und den Geschäftsprozessen fokussiert (Vgl. Halstenberg et al., (2020), S. 1).

*Electronic Data Interchange (EDI)*

EDI ist die Übertragung von Geschäftsdokumenten in einem standardisierten Datenformat zwischen verschiedenen Anwendungssystemen (Vgl. Kaufmann & Mülder, (2023), S. 99).

*Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)*

Unter EDIFACT wird ein einheitliches Regelwerk für den elektronischen Geschäftsverkehr verstanden. Dieses bezieht sich ausschließlich auf strukturierte Daten (Vgl. Kaufmann & Mülder, (2023), S. 99).

*Enterprise Resource Planning (ERP)*

ERP-Systeme sind große, modulare Unternehmens-Anwendungen, die für die meisten Geschäftsprozesse im Unternehmen bestimmt sind (Vgl. Pekša, (2018), S. 1).

Sie unterstützen ganzheitlich die Geschäftsprozesse in unterschiedlichen Funktionsbereichen des Unternehmens, wie zum Beispiel im Finanzwesen und Controlling, im Einkauf oder auch in der Produktionsplanung und Produktionssteuerung und in der Logistik. Dabei nutzen die verschiedenen Anwendungen des ERP-Systems eine gemeinsame Datenbasis (Vgl. Frick et al., (2008), S. 1).

*Extract‑Transform‑Load (ETL)*

Die Übernahme des operativen Datenbestands in den analytischen Datenbestand eines Data Warehouse bzw. eines Relational Database Management System (RDBMS) wird als ETL bezeichnet (Vgl. Kaufmann & Mülder, (2023), S. 380).

*Integrated Development Environment (IDE)*

Eine IDE (integrierte Entwicklungsumgebung) ist eine Sammlung von verschiedenen Werkzeugen, wie Editor, Compiler, Interpreter, Debugger, die für die Softwareentwicklung genutzt werden (Vgl. Kaufmann & Mülder, (2023), S. 503).

*Java Database Connectivity (JDBC)*

JDBC ist eine API zum Zugriff auf tabellarische Daten, insbesondere auf Daten in einem RDBMS (Vgl. ORACLE, (2023)).

*MySQL*

Die Open Source Software MySQL stellt sämtliche Funktionalitäten eines Relational Database Management System (RDBMS) zur Verfügung (Vgl. Burnus, (2008), S. 81).

*Open Source*

Die präzise Bezeichnung ist „Open-Source-Software“ und bezieht sich auf Software, deren Quellcode (Sourcecode) öffentlich zugänglich ist. Open-Source-Software darf ergänzt oder erweitert sowie kopiert und weitergegeben werden (Vgl. Kaufmann & Mülder, (2023), S. 283).

*Release Candidates (RC )*

Ein Software-Release ist ein (Software)-Produkt, das (an den Kunden) ausgeliefert wird. Release Candidate bezeichnet eine Softwareversion, die zur Auslieferung vorgesehen ist, aber noch nicht für den (End)-Anwender bereitgestellt wurde (Vgl. Kusay-Merkle, (2018), S. 292).

*Relational Database Management System (RDBMS )  
(relationales Datenbankverwaltungssystem)*

In einem RDBMS werden die Daten in zueinander in Relation stehenden Tabellen gespeichert (Vgl. Meier, (2010), S. 6).

*Requirements Engineering (Anforderungsmanagement)*

Das Requirements Engineering (RE) beinhaltet alle Tätigkeiten, die zur Erhebung und Analyse sowie dem Verständnis und zur Dokumentation der Anforderungen erforderlich sind (Vgl. Valentini et al., (2013), S. 9).

*Requirements Management (RM)*

Zum Requirements Management (RM) gehören alle für die Verwaltung und Bereitstellung und der Kommunikation von Anforderungen erforderlichen Tätigkeiten (Vgl. Valentini et al., (2013), S. 9).

*Repository*

Eine Bibliothek zur Verwaltung von Programmcode wird Repository genannt (Vgl. Kaufmann & Mülder, (2023), S. 502).

*Stakeholder*

Stakeholder sind Personen oder Personengruppen, die am Projekt interessiert oder beteiligt oder vom Projekt betroffen sind bzw. das Projekt beeinflussen können (Vgl. Timinger, (2015), S. 314 f.).

*Subversion (SVN)*

SVN ist ein Versionsverwaltungsprogramm, das eine Repository-Plattform für Programmierer zur Verfügung stellt, um gemeinsam an einem Softwareprojekt zu arbeiten (Vgl. Carle et al., (2013), S. 1).

*Validation*

Die Validation ist die Eignungsprüfung einer Systemkomponente bezogen auf ihren Einsatzzweck (Vgl. Alpar et al., (2023), S. 395).

*Verifikation*

Das Überprüfen der Übereinstimmung einer Systemkomponente mit ihrer Spezifikation wird als Verifikation bezeichnet (Vgl. Alpar et al., (2023), S. 398).

*Vorgehensmodell*

Das Vorgehensmodell umfasst alle Aktivitäten und deren Reihenfolge, die zur Durchführung eines Projektes erforderlich sind (Vgl. Aichele & Schönberger, (2014), S. 138).

*World Wide Web (WWW)*

Das World Wide Web (WWW) ist ein verteiltes Informationssystem und stellt Dienste für die Übertragung von Webseiten über das Hypertext Transfer Protocol (HTTP) zur Verfügung (Vgl. Kaufmann & Mülder, (2023), S. 214 f.).

*WWW-Server (Web-Server)*

Ein WWW -Server (Web-Server) ist eine Software, die Anfragen über im Internet über das HTTP-Protokoll empfängt und beantwortet (Vgl. Alpar et al., (2023), S. 537).

# Einleitung

Im Rahmen der Praxisphase des Bachelor Studiengangs „Wirtschaftsinformatik Online“ an der Hochschule für Technik Berlin wurde das Projekt zur Bereitstellung und Test der Software „Progress Monitor“ im Unternehmen Softzoll GmbH & Co. KG (Softzoll) durchgeführt.

Das operative Geschäft des Unternehmens besteht unter anderem in der Erbringung von Dienstleistungen im elektronischen Datenaustausch und der damit verbundenen Datenkonvertierung. Für die Realisierung der Dienstleistungen wird von Softzoll selbstentwickelte Software eingesetzt. Ferner sind für die Konvertierung Templates zu erstellen und Funktionsaufrufe zu parametrisieren. Die Templates und Funktionsaufrufe sowie die Einstellungen zum Datenaustausch werden durch die Mitarbeitenden in Workflows konfiguriert.

Zur Festigung bzw. Ausbau der Marktposition und der Wettbewerbsvorteile muss jedes Unternehmen in einem kontinuierlichen Verbesserungsprozess (KVP) die Produktqualität sowie die technischen und organisatorischen Prozesse im Unternehmen überprüfen und durch stetige Veränderungen verbessern (Vgl. Kirner et al., (2006), S. 2). Dazu gehört auch das Qualitätsmanagement, das sicherstellt, dass der Produktqualität erhalten bleibt bzw. die Produktqualität verbessert wird ((Vgl. Bechmann & Landerer, (2010), S. 9 ff.). Zur Fehleranalyse und zur stetigen Überprüfung der Softwarequalität sowie der Qualität der Arbeit wurde 2023 im Unternehmen die Software „Progress Monitor“ eingeführt.

Folgender Bericht dokumentiert die Einführung, Vorbereitung und Durchführung des Projekts.

# Methodik

Im methodischen Vorgehen gilt es zwischen der Vorgehensweise in der Projektdurchführung bzw. im Projektmanagement und in der Erstellung dieser Arbeit zu unterscheiden.

Zunächst erfolgt ein Einblick in die Methodik zur Erstellung der Arbeit. Die Erstellung dieser Arbeit beruht auf ein wissenschaftlich methodisches Vorgehen. In der Vorbereitung und beim Schreiben des Aufsatzes wurden die Methoden des Frameworks von Brocke et al angewandt. Die Methodik umfasst die fünf Phasen

* Definition des Rechercheumfangs
* Strukturierung und Aufbereitung des Themas (Konzeptualisierung)
* Durchführung der Literatur Recherche
* Analyse und Synthese der gefundenen Literatur
* Dokumentation der Forschungsergebnisse

(Vgl. Simons et al., (2009), S. 9).

Im ersten Schritt erfolgte die Abgrenzung der Themen und die Gliederung der Arbeit. Daran anschließend wurde das Grobkonzept für die Literaturrecherche und die Literaturanalyse sowie für die Artikelerstellung erstellt. Anhand der Gliederung und des Konzepts konnten die Kapitelüberschriften formuliert werden. In einer Mindmap wurden zu den Kapitelüberschriften die für die Literaturrecherche benötigten Suchbegriffe und sinnvolle Verknüpfungen der Suchbegriffe erfasst. Im nächsten Schritt erfolgte unter Verwendung der definierten Suchbegriffe die Literaturrecherche. In der Suche nach geeigneter Literatur ist der Zeitraum in der Regel bei wissenschaftlichen Artikeln auf fünf Jahre und bei themenbezogenen Fachbüchern auf fünfzehn Jahre beschränkt worden. Teilweise musste der Zeitraum auch erweitert werden, wenn die Suche nicht die gewünschten Resultate erbrachte. Um die Literaturrecherche später nachvollziehen zu können, wurde diese in einer Tabelle dokumentiert. Die Tabelle enthält zu jeder Recherche das Datum und die Uhrzeit, die verwendeten Suchbegriffe und Suchdatenbanken sowie den Titel der Literatur, den oder die Autoren, das Jahr der Publikation, den Verlag bzw. die Quelle der Publikation, die Art der Publikation (Journal oder Fachbuch oder Paper), die DOI‑ oder ISBN‑Nummer sowie die Anzahl der Zitate. Durch die Dokumentation der Literaturrecherche ist diese jederzeit reproduzierbar. Daraus ergibt sich, dass eine erneute Suche jederzeit dieselbe Literatur findet, sofern auf diese Artikel noch zugegriffen werden kann.

Durch das Lesen des Abstracts bzw. Vorworts wurde anschließend in der Analysephase die Literatur anhand der zuvor definierten Kriterien (Thema, Erscheinungsjahr, Quelle, fachliche Expertise) bewertet und selektiert. Die Literaturrecherche ist ein iterativer Prozess. Beim Anfertigen der Arbeit ergaben sich neue Fragen und die Anforderung weiterer Literatursichtung. Für die Recherche wurden die Datenbanken Google Scholar, Springer Link, „Webopac“ ‑ die Online-Bibliothek der BHT ‑ und Open Access genutzt. Mittels des Literaturverwaltungsprogramms Zotero ließ sich die Literatursammlung einfach verwalten und verschlagworten. Die Angabe der Literaturquellen in dieser Arbeit erfolgte ebenfalls mit Zotero .

Nach Timinger können im Projektmanagement je nach Anforderung verschiedene Arten von Vorgehensmodellen angewandt werden. Zum einen gibt es eine Differenzierung bezüglich des Einsatzbereiches. Darin wird zwischen allgemeinen und spezifischen Vorgehensmodellen, die nur für eine spezielle Branche, technische Lösung oder ein spezifisches Unternehmen angewandt werden, unterschieden. Eine weitere Unterscheidung besteht in der Art des Projektdurchführung bzw. Projektorganisation. Hier wird in sequenziellen, nebenläufigen bzw. parallelen, wiederholenden, agilen Vorgehensmodellen, prototypische und wiederverwendungsorientierte Vorgehensmodellen unterschieden. Beim sequenziellen Vorgehen wird das Projekt in einzelnen Phasen, die nacheinander abgearbeitet werden, untereilt. Jede Phase muss abgeschlossen sein, bevor die folgende Phase begonnen wird. Das Wasserfallmodell ist eines der bekanntesten sequenziellen Vorgehensmodelle. Es beinhaltet die Phasen:

* Anforderungsphase
* Analysephase
* Entwurfsphase / Spezifikation
* Implementierungsphase
* Testphase
* Transferphase

Dieses Vorgehensmodell ermöglicht während des Projektverlaufs den Rücksprung in eine vorausgegangene Phase (Vgl. Timinger, (2015), S. 71 ff.).

In dem hier betrachteten Projekt wurden die genannten Phasen nach dem Wasserfallmodell durchlaufen.

# Unternehmensprofil der „Softzoll GmbH & Co. KG“

Die Softzoll GmbH & Co. KG (Softzoll) ist ein mittelständisches Unternehmen in der IT‑Branche mit den Standorten in Berlin und Göttingen. Als IT‑Dienstleister unterstützt das Unternehmen seine Kunden im elektronischen Datenaustausch (EDI) und in der damit verbundenen Datenkonvertierung. Die mit dem EDI verbundenen Dienste werden dabei zentral in einer Cloud sowie auch dezentral im IT‑Netzwerk des Kunden erbracht. Für den Nachrichtenaustausch und die Datenkonvertierung kommen verschiedene Softwarekomponenten als Schnittstelle zwischen den Kunden des Unternehmens und deren Partner zum Einsatz. Der Austausch der Daten zwischen den Partnern erfolgt über verschiedene Kommunikationskanäle und Netzwerkprotokolle, wie Applicability Statement 2 (AS2), File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), Hypertext Transfer Protocol Secure (HTTPS), Secure File Transfer Protocol (SFTP), Simple Mail Transfer Protocol (SMTP), X.400.

Zum Leistungsportfolio gehören auch die Analyse der Ist‑Situation und Geschäftsprozesse beim Kunden, die Anforderungsanalyse, die Beratung und selbstverständlich auch die Schulung der Anwender im Umgang mit der für den Nachrichtenaustausch und die Konvertierung genutzten Software.

Zum Kundenklientel gehören Kunden aus den Branchen Automotive, Handel, Logistik, Maschinenbau, Produktion, Elektronik und weitere zusammen.

Die Abteilungen der Softzoll sind hierarchisch in einer Linienorganisation strukturiert. Dabei sind die Aufgaben- und Verantwortungsbereiche in den Abteilungen klar voneinander abgegrenzt. Aus der Linienorganisation ergeben sich auch strikte Weisungsbefugnis und der Berichtspfad.

In den folgenden Kapiteln werden die Tätigkeiten zur Projektvorbereitung, wie die Projektbegründung, die Definition des Projektumfangs und des Projektziels beschrieben.

# Projektbegründung

Vor dem Start des Projekts wurden das Projektziel formuliert, der Untersuchungs‑ und Aufgabenbereich zum Projekt abgegrenzt und basierend auf die Probleme definiert. Aus der Problembeschreibung ließen sich die konkreten Anforderungen ableiten und die sich daraus ableitenden Aufgaben spezifizieren. Zu den Aufgaben wurden die für die Bearbeitung erforderlichen Ressourcen bestimmt. Anschließend ließ sich der zeitliche Rahmen bestimmen und eine grobe Kostenkalkulation aufstellen. Der Abgleich der kalkulierten Kosten mit dem Projektbudget zeigte, dass das Projekt im verfügbaren Budget realisierbar ist. Anhand der Ergebnisse aus der ersten Phase der Projektbegründung basierte die daran anschließende Machbarkeitsstudie, in deren Verlauf erst eruiert werden konnte, ob die Anforderungen mit den verfügbaren Ressourcen und im Zeit‑ und Kostenrahmen erfüllt werden können und das Projektziel erreicht werden kann. Im nächsten Schritt erfolgte eine grobe Projektorganisation. Diese beinhaltete einen groben Projektplan mit Meilensteinen (Termine für Etappenziele im Projekt), die im Projekt involvierten Stakeholder, die Zuordnung der Ressourcen zu den Aufgaben, Verantwortlichkeiten, Randbedingungen.

Aus den Ergebnissen der Projektbegründung wurden der Projektantrag und der dazugehörige Projektauftrag erstellt.

Nach Freigabe des Projektantrags konnte mit der eigentlichen Durchführung des Projekts begonnen werden.

## Projektziel und Projektumfang

Das Ziel des unternehmensinternen Projekts ist die Entwicklung, Integration und Bereitstellung der Software „Progress Monitor“, die der Beobachtung der Datenkonvertierung und der Analyse von Fehlern, die in der Konvertierung auftreten können, dient. Ferner soll die zu entwickelnde Software die Konfiguration von Workflows, in denen die für die Datenkonvertierungen und Kommunikation notwendigen Konfigurationen gespeichert werden, ermöglichen und die mitarbeitenden im technischen Support unterstützen. Details der Anforderungen sind im Kapitel Anforderungsanalyse“ beschrieben.

Die Projektdurchführung beinhaltet Tätigkeiten des Projektmanagements, wie die Projektvorbereitung einschließlich der Aufgaben aus der Projektbegründung, die Kalkulation, die Projektorganisation und Projektsteuerung und Tätigkeiten zur Projektrealisierung. Dazu gehören die Anforderungs- und Risikoanalyse, die Erhebung der Ist‑Situation, das Erstellen des Soll‑Konzepts sowie die Implementierung, die Integration und Tests. In der Implementierung wurde die Software nach dem DevOps‑Ansatz entwickelt, getestet und bereitgestellt. Dieser erlaubt eine kurzfristige Anpassung der Software auf geänderte bzw. erweiterte Anforderungen und einen kontinuierlichen Verbesserungsprozess (KVP), an dem sich die Unternehmensstrategie und die Planung und das Handeln des Unternehmens ausrichtet. Die Schulung der Anwender und die Softwarewartung sind die Tätigkeiten der letzten Phase des Phasenmodells.

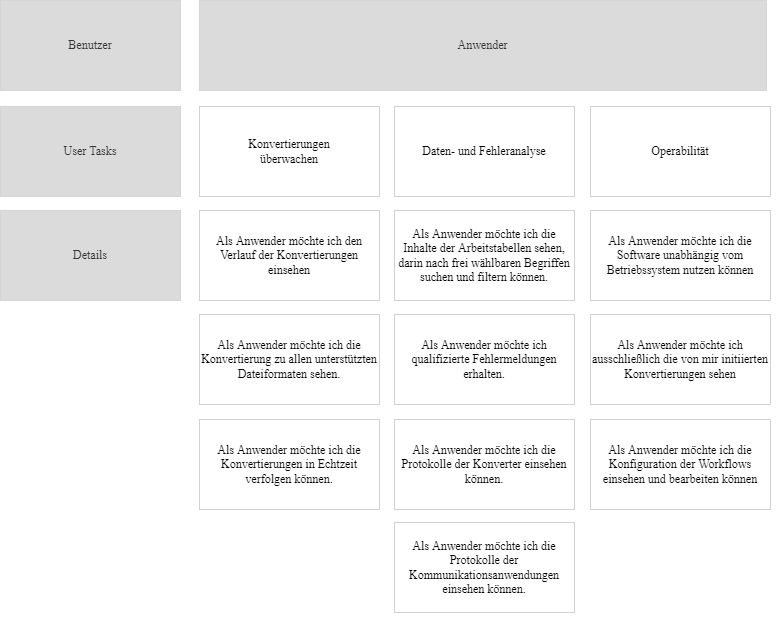
An dem Projekt beteiligte Stakeholder waren die Mitarbeitenden der Abteilungen „Geschäftsleitung“, „Change‑ und Qualitätsmanagements“, „Controlling“ und „Softwareentwicklung“ als Verantwortliche für die Durchführung sowie die Mitarbeitenden der Abteilung „EDI‑Projektmanagement“ als Kunden.

# Anforderungsanalyse (Requirements Engineering)

Die Definition der Anforderungen (Requirements) ist eine notwendige Grundlage für die Entwickelung von IT‑Systemen. Die präzise Beschreibung der zu erfüllenden Anforderungen ermöglicht die Erfüllung der Benutzeranforderungen und die Erreichung des Projektziels im geplanten Kostenrahmen und Zeitraum (Vgl. Herrmann, (2022), S. 16 ff.).

Als innovative Technik in der agilen Prozess‑ und Anforderungsanalyse hat sich das Story‑Mapping etabliert, Darin werden unter der Beteiligung aller Stakeholder die Anforderungen der Benutzer in User Storys erfasst. Die Anforderungsanalyse ist, wie die Entwicklung der Anwendung ein iterativer Prozess (Vgl. Herrmann, (2022), S. 18).

Formulierte Anforderungen in den User Story werden im weiteren Verlauf mit dem Kunden diskutiert bis für den Kunden und den Auftragnehmer ein einheitliches Verständnis und eine Einigung über jede einzelne Anforderung besteht. In der Anforderungsanalyse erfolgt ferner eine Differenzierung der Anforderungen in funktionale und nicht funktionale Anforderungen. Nachfolgende Abbildung zeigt exemplarisch einen Auszug aus der zu diesem Projekt erstellten User Story Map.



(Quelle: Softzoll GmbH & Co. KG, Story Map, 2023)

Abb. 5.1: User Story Map

Im Anhang können der Tabelle „Anforderungen aus dem Anforderungsdokument ReqSpec 1.001 vom 26.06.2023“[[1]](#footnote-1) die im Projekt identifizierten Anforderungen entnommen werden.

## Risikoanalyse (Projekt Risikomanagement)

Aus den typischen Merkmalen eines Projekts ‑ hohe Komplexität, festgelegter Umfang, Interdisziplinarität, Einmaligkeit, zeitliche, finanzielle, personelle oder andere Begrenzungen – resultieren auch Risiken in der Projektdurchführung (Vgl. Wack, (2007), S. 5). In der Literatur wird der Risikobegriff unterschiedlich definiert (Vgl. Wack, (2007), S. 19 ff.). Daraus ergibt sich, dass im Projektmanagement verschiedene Risikofelder zu beachten sind (Vgl. Wack, (2007), S. 43).

Aufgrund der Einmaligkeit können die Projektverantwortlichen nur auf einen begrenzten Erfahrungsschatz zugreifen. Für die erfolgreiche Realisierung von IT-Projekten ist ein Projekt‑Risikomanagement notwendig. Im Projekt‑Risikomanagement werden potentielle Risiken während des Projektverlaufs fortlaufend identifiziert und bewertet und die Risikoauswirkungen durch geeignete Maßnahmen minimiert. Im Monitoring‑Prozess wird Umsetzung und der Erfolg der Umsetzung der Maßnahmen überwacht. Zur Aufgabe der für das Risikomanagement Verantwortlichen gehört auch aus den Ergebnissen des Monitoring die Wirksamkeit der Maßnahmen zu beurteilen und ggf. die Maßnahmen anzupassen (Vgl. Wack, (2007), S. 29 ff.).

Zur Festlegung von Maßnahmen bei Eintritt eines Risikofalls werden die Risiken nach der Identifizierung hinsichtlich ihres Schadensausmaßes, der Eintrittswahrscheinlichkeit (Ewk) und der Auswirkung bewertet. Aus dieser Analyse wird für jedes identifizierte Risiko entschieden in welchem Umfang Maßnahmen entwickelt werden. Die Bewertung kann auch ergeben, dass eine Maßnahme im Verhältnis zur Ewk oder zum Schadensausmaß bzw. zur Auswirkung unwirtschaftlich ist und daher zum Risiko keine Maßnahme festgelegt wird (Vgl. Wack, (2007), S. 23 f.).

Die Prozesse im Projekt‑Risikomanagement werden während des gesamten IT-Projekts iterativ durchlaufen, da sich in der Projektdurchführung die Auswirkung, Ewk und das Schadensausmaß zu identifizierten Risiken ändern bzw. neue Risiken identifiziert werden können. Daraus resultieren dann Anpassungen der Maßnahmen oder die Entwicklung neuer Maßnahmen.

In diesem Projekt wurden während der Projektdurchführung im Projekt‑Risikomanagement gemeinsam mit den Stakeholdern kontinuierlich die Risiken identifiziert, analysiert und bewertet sowie geeignete Maßnahmen zur Reduzierung des Schadensausmaßes festgelegt. Das Ergebnis der Risikoanalyse wurde in dem im Anhang beigefügten Risikokatalog[[2]](#footnote-2) dokumentiert.

# Ist-Aufnahme im Unternehmen Softzoll GmbH & Co. KG

In der Phase der Projektbegründung und Definition des Projektziels und Projektumfangs wurden bereits das Projektziel und der Projektumfang, die Stakeholder und der Untersuchungsbereich festgelegt.

Während der Ist-Aufnahme gilt es, die Geschäftsprozesse sowie bestehende IT-Systeme und Schnittstellen zu diesen Systemen zu identifizieren und zu analysieren, die für den Untersuchungsbereich relevant sind.

Zur Aufnahme der Geschäftsprozesse können Methoden wie die Befragung, Beobachtung oder die Ermittlung von Betriebsdaten (dokumentenzentrierte Technik) eingesetzt werden (Vgl. SOPHIST GmbH & Rupp, (2013), S. 150, Krallmann et al., (2013), S. 145 ff).

In der Durchführung dieses Projekts erfolgte die Ist-Aufnahme der Geschäftsprozesse über die Befragung der Anwender und Beobachtung. In der Beobachtung wurden im ersten Schritt die Geschäftsprozesse und dazugehörigen Teilprozesse aufgenommen und im zweiten Schritt die zu konvertierenden Dateien analysiert. Die Ergebnisse der Analyse wurden im Projektbericht dokumentiert. Folgend wird der Geschäftsprozess zur Daten-Konvertierung und die dazugehörigen Unterprozesse beschrieben.

Die Datenkonvertierung basiert auf dem Extract-Transform-Load-Ansatz (ETL). ETL ist ein drei-stufiger Prozess in welchem die Daten aus einer Datenquelle (in der Regel eine Datei) extrahiert und in eine Datenbank geschrieben, anschließend transformiert und im letzten Schritt in ein Ziel-Datenobjekt (in der Regel eine Datei) geladen werden. Im Transformationsprozess erfolgt die Aufbereitung der Daten durch Transformationsregeln entsprechend der durch den Geschäftsprozess spezifizierten Anforderungen. Dabei werden Daten entfernt oder um von zusätzliche Informationen angereichert, Datums‑ oder Zahlenwerten formatiert, Werte berechnet, Daten normalisiert, fehlende Einheiten ergänzt oder Einheiten umgerechnet.

Ein Konvertierungsprozess besteht aus mehreren Unterprozessen. Jeder Unterprozess erfüllt eine bestimmte Funktion, wie den Empfang von Nachrichten, das Extrahieren der Daten aus einer Datenquelle, die Transformation, das Schreiben der Daten in ein Datenobjekt, den Versand der Daten, die Überwachung des IT-Systems, den Versand von Nachrichten zu Fehlermeldungen. Für die Konvertierung kommen pro Datentyp dedizierte Komponenten zur Anwendung. Durch den modularen Aufbau des zur EDIFACT-Konvertierung genutzten IT-Systems (Daten-Konverter) in Form einzelner Software-Komponenten ist jeder einzelne Unterprozess gekapselt.

Das erleichtert die Entwicklung neuer Funktionalitäten, da diese spezifisch für einen bestimmten Prozess entwickelt werden können. Ferner hat ein Fehler in einer Konvertierung keine Auswirkungen auf andere Konvertierungen.

Die Konvertierungen basieren auf Workflows, d. h. automatisierten Geschäftsprozessen. Jeder Workflow ist mit einem Anwendungsfall (Use Case) verknüpft. Im Use Case ist definiert, welcher Kunde mit welchem Geschäftspartner welche Art von Daten (in EDI Nachrichten genannt) austauscht. Daraus resultiert, dass für jeden Anwendungsfall (Use Case) genau ein Workflow eingerichtet ist.

Im Workflow sind Templates für das Schreiben der Daten in die Datenbank nach dem Extrahieren sowie für die Transformation und den Datenexport in das Ziel-Datenobjekt hinterlegt. Für die Konvertierung wird für jeden Datentyp ein Konverter und daraus folgend ein eigener Template-Typ verwendet. So kommt bei der Konvertierung von XML-Dateien ein anderer Konverter bzw. ein anderer Template-Typ als bei der Konvertierung von CSV-Dateien zum Einsatz. Ferner enthält der Workflow Parameter zum Anwendungsfall, wie Nachrichtentyp, Sender-ID, Empfänger-ID, Parameter zur Kommunikationsanbindung. Eine Konfigurationsdatei im XML-Format ist die Schnittstelle zwischen Daten, die beim Extrahieren aus der Datenquelle ausgelesen werden und den Workflow-Parametern. Anhand dieser Workflow-Parameter wird der Workflow identifiziert und die im Workflow hinterlegten Templates verwendet.

In der Konvertierung wird aus Sicht des Daten-Konverters in ein- und ausgehende Daten unterschieden. Eingehende Daten sind vom Geschäftspartner des Kunden, die nach der Konvertierung an den Kunden gesandt werden. Bei ausgehenden Daten empfängt der Daten-Konverter die Daten vom Kunden und leitet die nach der Konvertierung erstellte Datei an den Geschäftspartner des Kunden weiter. Der Ablauf der Konvertierung unterscheidet sich bei ein‑ und ausgehenden Daten nicht und stellt sich wie folgt dar.

Der Ablauf der Konvertierung ist wie folgt:

Die Daten werden vom Geschäftspartner des Kunden oder vom Kunden über verschiedene Kommunikationswege an den Daten-Konverter gesandt. Anschließend wird die Datei geöffnet und ausgelesen. Anhand der in der Konfigurationsdatei definierten Schemata wird in den Workflow-Konfigurationen nach passenden Parametern gesucht. Dabei werden basierend auf den Schemata in der Konfigurationsdatei Muster aus den eingelesenen Daten mit den Workflow-Konfigurationen verglichen. Dieser Vergleich basiert auf einer SQL-Abfrage. Wird ein passender Workflow identifiziert, werden die Daten aus der Datenquelle in eine MySQL-Datenbank geschrieben. Die Zuordnung der Daten zu den dafür entsprechenden Datenbankfeldern wird über ein im Workflow hinterlegten Template realisiert. Nachdem die Daten in die MySQL-Datenbank geschrieben wurden, wird der Transformationsprozess ausgeführt. Im letzten Schritt werden die transformierten Daten im Load-Prozess in ein Datenobjekt geschrieben. Der Versand der konvertierten Daten an den Kunden bzw. den Geschäftspartner des Kunden nach Abschluss der Konvertierung schließt den Geschäftsprozess ab. Jeder Unterprozess in der Konvertierung wird durch den Daten-Konverter überwacht. Wenn Fehler einen Prozess unterbrechen, wird über das Kommunikationsprotokoll SMTP eine Nachricht an festgelegte Empfänger versandt. Diese Nachricht enthält Informationen wie einen Transaktionsidentifier, einen Datums-/Zeitstempel, den Workflow-Identifier, den Workflownamen, einen Fehlercode und eine Fehlerbeschreibung. Fehlercode und Fehlerbeschreibung sind dabei vom Fehlertyp abhängig. Das Routing von Daten ist ein besonderer Anwendungsfall. Dabei werden die Daten nicht konvertiert und direkt an den im Workflow hinterlegten Empfänger weitergeleitet.

Folgende Tabelle listet die möglichen Datenformate, die konvertiert werden können und die möglichen Kommunikationsprotokolle auf.

Tab. .: Übersicht zu unterstützten Datenformaten und Kommunikationsprotokollen

|  |  |  |
| --- | --- | --- |
| Richtung | Datenformat | Kommunikationsprotokoll |
| eingehend | Comma Separated Value (CSV)  Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)  Fix Format (FF)  Hypertext Markup Language (HTML)  Intermediate Document (IDoc)  JD Edwards (JDE)  Portable Document Format (PDF)  Tradacoms (TRC)  ANSI X12 (X12)  Extensible Markup Language (XML) | Applicability Statement 2 (AS2)  File Transfer Protocol (FTP)  Hypertext Transfer Protocol (HTTP)  Hypertext Transfer Protocol Secure (HTTPS)  Odette File Transfer Protocol (OFTP)  Secure File Transfer Protocol (SFTP)  Simple Mail Transfer Protocol (SMTP)  X.400 |
| ausgehend | Comma Separated Value (CSV)  Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT)  Fix Format (FF)  Hypertext Markup Language (HTML)  Intermediate Document (IDoc)  JD Edwards (JDE)  Portable Document Format (PDF)  Tradacoms (TRC)  ANSI X12 (X12)  Extensible Markup Language (XML) | Applicability Statement 2 (AS2)  File Transfer Protocol (FTP)  Hypertext Transfer Protocol (HTTP)  Hypertext Transfer Protocol Secure (HTTPS)  Odette File Transfer Protocol (OFTP)  Secure File Transfer Protocol (SFTP)  Simple Mail Transfer Protocol (SMTP)  X.400 |

Die folgende Geschäftsprozesse in BPMN dient zur übersichtlicheren Darstellung und zum besseren Verständnis.



(Quelle: Softzoll GmbH & Co. KG, BPMN-Diagramm zum Prozess „Datenkonvertierung“, 2023)

Abb. 6.1: BPMN-Diagramm zum Prozess "Datenkonvertierung"

Neben der Analyse der Geschäftsprozesse sind in der Ist-Aufnahme die zum Untersuchungsbereich gehörenden IT-Systeme und Schnittstellen zwischen den IT-Systemen zu erfassen (Vgl. Krallmann et al., (2013), S. 145 ff.).

Darin wurden folgende IT-Komponenten und deren Schnittstellen identifiziert.

Tab. 6.2: Übersicht zu den existierenden Systemkomponenten

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Komponente | Beschreibung | | Informationen zur Implementierung | | Schnittstelle |
| AS2-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende AS2-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| FTP-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende FTP-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| HTTP-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende HTTP-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| HTTPS-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende HTTPS-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| OFTP-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende OFTP-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| SFTP-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende SFTP-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| SMTP-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende SMTP-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| X.400-Kommunikations-  dispatcher | Komponente für die ein- und ausgehende X.400-Kommunikation | | entwickelt in C++  basiert auf QT-Framework | | Daten-Konverter |
| CSV-Converter | | Komponente für die Konvertierung von CSV-Dateien | | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| FF-Converter | | Komponente für die Konvertierung von FF-Dateien | | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| EDIFACT-Converter | | Komponente für die Konvertierung von EDIFACT-Dateien | | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |

|  |  |  |  |
| --- | --- | --- | --- |
| Komponente | Beschreibung | Informationen zur Implementierung | Schnittstelle |
| HTML-Converter | Komponente für die Konvertierung von HTML-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| IDoc-Converter | Komponente für die Konvertierung von SAP IDoc-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| JDE-Converter | Komponente für die Konvertierung von JDE-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| PDF-Converter | Komponente für die Konvertierung von PDF-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| TRC-Converter | Komponente für die Konvertierung von TRC-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| X.12-Converter | Komponente für die Konvertierung von X.12-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| XML-Converter | Komponente für die Konvertierung von XML-Dateien | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| Scheduler | Komponente für die zeitgesteuerte Konvertierung | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| Dispatcher-Check-System | Komponente für die Systemüberwachung | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| Dispatcher-Workflow-Erkennung | Komponente für die Identifizierung der Workflows | entwickelt in C++  basiert auf QT-Framework | ODBC (MySQL-DB) |
| MySQL-DB | RDBMS | Datenbank für die Datenhaltung |  |

Das Komponenten-Diagramm veranschaulicht die in der Ist-Aufnahme identifizierten Systemkomponenten.



(Quelle: Softzoll GmbH & Co. KG, BPMN-Diagramm zum Prozess „Datenkonvertierung“, 2023)

Abb. 6.2: Komponentendiagramm zu den identifizierten Systemkomponenten

# Entwurf / Spezifikation

In der Entwurfsphase werden die in der Anforderungsanalyse identifizierten Anforderungen in ein in ein Lösungskonzept umgesetzt. Die Ergebnisse der Entwurfsphase sind z. B. Lastenheft, und Systemspezifikation. Darin werden die Eigenschaften der zu entwickelnden Software beschrieben und im weiteren Verlauf systematisch konkretisiert. In den Entwurfsentscheidungen werden die Ergebnisse zu den beschriebenen Eigenschaften umgesetzt (Vgl. Beifuss & Holzbaur, (2020), S. 72).

Dabei werden die Analysemodelle und Anforderungen sukzessive verfeinert, ergänzt und konkretisiert. Die Ergebnisse der Spezifikation und des Entwurfs dienen als Vorgabe zur Software-Implementierung. In den Entwurfsentscheidungen sind die GUI, die Programmiersprache, das Datenbanksystem, mögliche Verteilungskonzepte, Komponentenwiederverwendung und andere Aspekte zu berücksichtigen. Die Entwurfsmodelle müssen auf einem solchen Niveau sein, dass die Implementierung eine signifikante Verfeinerung des Entwurfs darstellt (Vgl. Alpar et al., (2023), S. 458).

Diese Methodik für das Projektmanagement in der Softwareentwicklung beschreibt auch die Tätigkeiten, die in der Entwurfs‑ und Spezifikationsphase ausgeführt wurden. Anhand der in der identifizierten Anforderungen wurden die Eigenschaften der zu entwickelnden Software spezifiziert und das Konzept sowie die Entwurfsentscheidungen erarbeitet. Die wichtigsten Entwurfsentscheidungen sind folgender Tabelle zusammengefasst.

Tab. 7.1: Spezifikation und Entwurfsentscheidungen

|  |  |  |  |
| --- | --- | --- | --- |
| Entwurf / Spezifikation zur Software „Progress Monitor“ | | | |
| Zweck:  Überwachung der Datenkonvertierungen  Analyse der Datenkonvertierungen  Fehleranalyse  Unterstützung im fachlichen und technischen Support  Qualitätssicherung, Qualitätsverbesserung  Schulung von Kunden und Mitarbeitenden | | | |
| Software | | | |
| Nr. | Spezifikation | Anforderung |
|  | Die Software wird für die aktuelle Version der Java Runtime Environment (JRE) entwickelt | Die Software muss unabhängig vom Betriebssystem nutzbar sein |

|  |  |  |
| --- | --- | --- |
| Software | | |
| Nr. | Spezifikation | Anforderung |
|  | Das zugrundeliegende RDBMS ist MySQL, Version 5.0.45-community-nt | Nutzung einer gemeinsamen Datenbasis für den Daten-Konverter und die Software  „Progress Monitor“ |
|  | Als Webserver wird Apache Tomcat eingesetzt. | Für den Webserver soll etablierte Open-Source-Software eingesetzt werden |
|  | Die Schnittstelle zwischen der Software und der Datenbank wird über Java Database Connectivity, Version 4.3 (JDBC) realisiert | Nutzung einer gemeinsamen Datenbasis für den Daten-Konverter und die Software „Progress Monitor“ |
|  | Die Ergebnisse der Konvertierung werden in der Software über Tabellen dargestellt. | Der Anwender muss die Ergebnisse zur Konvertierung filtern und sortieren können. |
|  | Die Tabellen in der Software sind sortierbar und filterbar. | Der Anwender muss die Ergebnisse zur Konvertierung filtern und sortieren können. |
|  | Entwicklung einer Benutzerkontenverwaltung | Verwaltung von Zugriffsberechtigung aus Sicherheitsgründen |
|  | Entwicklung von Benutzerrollen und einer Zugriffssteuerung mit dedizierten Benutzerrechten | Steuerung von dedizierten Benutzerrechten aus Sicherheitsgründen |
|  | Implementierung einer Mandantenverwaltung | individuelle Zugriffssteuerung pro Mandant und Benutzer |
|  | Implementierung zur Protokollierung der Benutzeraktionen | Sämtlich Benutzeraktionen müssen zur Nachvollziehbarkeit protokolliert werden. |
|  | Die Anwendung ist für die Anwender über die Browser Google Chrome (ab Version ), Firefox (ab Version ), Microsoft Edge (ab Version ) nutzbar | Die Software muss unabhängig vom Betriebssystem nutzbar sein |
|  | Die Kommunikation mit dem Webserver, welcher den progress Monitor bereitstellt, wird mittels SSL verschlüsselt und über das Anwendungsprotokoll HTTPS realisiert.  Für den Web-Server darf nur in von einer beglaubigten Zertifizierungsstelle ausgestelltes Zertifikat bereitgestellt werden. | Aus Sicherheitsgründen muss die Kommunikation zwischen Endgerät und Anwendung über SSL verschlüsselt werden. |
|  | Die im Frontend präsentierten Webseiten werden mittels HTML erstellt.  Die Gestaltung (Formatierung) dieser Webseiten erfolgt ausschließlich über Cascading Stylesheets (CSS).  Im Frontend zu realisierende Funktionalitäten werden mittels Java-Script umgesetzt. | Für die Präsentation der Websites sind etablierte und standardisierte Auszeichnungssprachen und Werkzeuge zu verwenden. |

|  |  |  |
| --- | --- | --- |
| Softwareentwicklung | | |
| Nr. | Spezifikation | Anforderung |
|  | Die Software wird nach dem DevOps-Konzept mit den darin integrierten Techniken  Kontinuierliche Optimierung,  Releaseplanung (Release)  Kontinuierliche Integration (Continuos Integration)  Kontinuierliche Bereitstellung (Continuos Deployment)  entwickelt | Durch die Anwendung des DevOps-Konzepts ist die Zusammenarbeit der Funktionsbereiche „Softwareentwicklung“ und in „IT-Operations“ zu intensivieren und infolgedessen die Effektivität und Effizienz der Arbeit zu verbessern.  Ziel der Umsetzung des DevOps-Konzepts ist die Erhöhung des Betriebsergebnisses und die Qualitätssicherung in der Softwareentwicklung als Baustein des kontinuierlichen Verbesserungsprozesses. Durch das DevOps-Konzept sollen in der Implementierungsphase frühzeitig funktionsfähige Prototypen bereitgestellt werden. Das Feedback der Anwender zu den Prototypen soll in neue Release-Versionen einfließen. |
|  | Die Software wird in der integrierten Entwicklungsumgebung (Integrated Development Environment (IDE)) Eclipse entwickelt. | Die Software soll in einer etablierten Open-Source Entwicklungsumgebung, in welcher die Anbindung an das Versionsverwaltungssystem „Subversion“ (SVN) integriert ist, und die den Softwareentwicklern vertraut ist, entwickelt werden. |
|  | Änderungen an der Software werden als Builds zentral in dem Versionsverwaltungssystem SVN verwaltet. | Software-Builds und -Releases sind in der Versionsverwaltung (Repository-Verwaltung) SVN zentral zu organisieren und zu archivieren.  Die zentrale Verwaltung verschiedener Versionsstände in Forks und Branches sichert jeden Versionsstand und ermöglicht, etwa im Fehlerfall, die Wiederherstellung früherer Versionsstände in der Versionshistorie. Durch die Protokollierung und Archivierung lassen sich Änderungen nachvollziehen. Die Implementierung basierend auf SVN ist ein zentraler Bestandteil der Qualitätssicherung bei Softzoll. |
|  | In der Softwareentwicklung werden zur Qualitätssicherung Metrik-Werkzeuge wie FindBugs, JDepend, Checkstyle integriert.  Metrikregeln, die der Sicherstellung der Quellcodequalität dienen, werden vor jedem Build automatisiert ausgeführt. Entspricht der Quellcode nicht den in den Regeln definierten Anwendungen wird der Build-Prozess abgebrochen. | Für die Gewährleistung der Software-Qualität sind in der Software-Entwicklung geeignete Metrik-Werkzeuge wie FindBugs, JDepend, Checkstyle regelmäßig zu nutzen.  Der Sicherstellung der Code-Qualität, wie die Bezeichnung von Klassen, Methoden, Variablen, Konstanten, die Kommentierung und Formatierung des Quellcodes, ist dabei die größte Aufmerksamkeit zu widmen. |

|  |  |  |
| --- | --- | --- |
| Softwareentwicklung | | |
| Nr. | Spezifikation | Anforderung |
|  | In der Softwareentwicklung wird zur Qualitätssicherung regelmäßig das Refactoring zur Quellcode-Refrakturierung angewandt. | Für die Gewährleistung der Software-Qualität ist der Quellcode regelmäßig zu refrakturieren. |
|  | Zur Überprüfung der Ausführung und Korrektheit wird der Quellcode kontinuierlich und automatisiert durch Modultests (Unit Tests) validiert und verifiziert. | Der Quellcode ist kontinuierlich und automatisiert nach den Vorgaben im Validation- und Verification-Plan durch Modultests (Unit Tests) zu validieren und verifizieren.  Der Validation- und Verification-Plan ist ein essentieller Bestandteil der Qualitätssicherung. |
|  | Die Ausführung und korrekte Umsetzung der geforderten Funktionalitäten in der Software wird kontinuierlich durch Produkttests und Akzeptanztests verifiziert. | Die korrekte Umsetzung der geforderten Funktionalitäten ist regelmäßig durch Produkttests und Akzeptanztests, die nach den Vorgaben im Validation- und Verification-Plan ausgeführt werden, nachzuweisen. Der Validation- und Verification-Plan ist ein essentieller Bestandteil der Qualitätssicherung. |

# Implementierung

Das Projektmanagement und die Durchführung der Integrations- und Systemtests sowie die Bereitstellung der Anwendung „Progress Monitor“ stehen in diesem Artikel im Vordergrund. In der Projektdurchführung hat der Autor des Artikels eng mit der Softwareentwicklung zusammengearbeitet, jedoch die Software nicht programmiert. Zur vollständigen Beschreibung der Projektdurchführung und als Einleitung für die folgenden Kapitel wird die Umsetzung der Implementierung im folgenden Absatz zusammengefasst.

Die Implementierung erfolgte nach dem Dev-Ops-Konzept. „Dev-Ops“ setzt sich aus den Worten „Development“ und „(IT)-Operations“ zusammen und stellt die Zusammenarbeit zwischen diesen beiden Funktionsbereichen im Unternehmen in den Vordergrund. Das DevOps-Konzept beinhaltet praxiserprobte Lösungsansätze um Lücken zwischen den Softwareentwicklung und dem IT-Betrieb zu überbrücken (Vgl. Halstenberg et al., (2020), S. 1). In einem Dev-Ops-Zyklus werden die Phasen „Plan“, „Code“, „Build“, „Integrate“, „Release“ und „Operate“ durchlaufen. Während der Release-Phase werden die Modultests (Unit-Test) und die Bereitstellung der Software (Deploy) realisiert. Aktivitäten, in den Schritten „Build“, „Integrate“, „Test“ und „Deploy“ können weitestgehend automatisiert werden. Damit deckt der Dev-Ops-Zyklus den gesamten Software-Lebenszyklus ab. Dev-Ops basiert auf dem Prinzip der kontinuierlichen Integration („Continuous Integration“) und kontinuierliche Bereitstellung („Continuous Deployment“). Eine vollständige Automatisierung der Integration, Erstellung und Tests ermöglicht die automatische Bereitstellung eines fertig getesteten Release Kandidaten(Vgl. Halstenberg et al., (2020), S. 16 ff.).

In der IDE „Eclipse“ erstellten die Softwareentwickler den Programmcode, der in der Programmiersprache „Java“ geschrieben ist. Die zentrale Verwaltung der Versionierung erfolgte durch SVN. Mittels automatisierte Modultests (Unit Tests) wurde die Ausführung und Korrektheit des Programmcodes validiert und verifiziert.

Als zentrales RDBMS diente der MySQL-Datenbankserver und für die Bereitstellung der HTML-Seiten der Web-Server Apache Tomcat.

Das folgende Kapitel beschäftigt sich detailliert mit den Integrations- und Systemtests.

# Test

## Einführung

Zunächst wird die Frage nach dem Zweck von Software-Tests beantwortet. Software-Tests generieren keinen ökonomischen Ertrag, aber dennoch einen ökonomischen Nutzen. Sie liefern eine Definition der Qualitätskriterien, decken Mängel bzw. Fehler auf und motivieren die Softwareentwickler gute Software zu entwickeln. Durch das frühzeitige Aufdecken von Mängeln bzw. Fehlern werden Folgekosten, die aus Nachbesserung oder Schadensersatzforderungen resultieren, gespart (Vgl. Frühauf et al., (2007), S. 14, 29).

Mit der Implementierung starten auch die Software-Tests. Diese umfassen analytische Maßnahmen zur Sicherstellung der Softwarequalität. Software-Tests sind in allen Phasen des Entwicklungsprozesses durchzuführen (Vgl. Hoffmann, (2013), S. 158). Dabei werden im ersten Schritt die einzelnen Softwarekomponenten getestet. Den Komponententest schließen sich die Tests einzelner Teilsysteme und im letzten Schritt die Tests des Gesamtsystems an (Vgl. Alpar et al., (2023), S. 469 f.).



(Quelle: Alpar et al., 2023, S. 470)

Abb. 9.1: Grundlegende Teststrategie

In den Tests werden das Verhalten der Software und die Erfüllung der Spezifikation bzw. des Entwurfs (Verifikation) sowie die Erfüllung der Anforderungen im Nutzungskontext (Validation) geprüft. Die Testaktivitäten werden aus den Anforderungen abgeleitet.

Die Tests lassen sich nach ihren Merkmalen in die Klassen „Prüfebene - In welcher Entwicklungsphase wird getestet?“, „Prüfkriterium - Welche Inhalte werden getestet?“, „Prüfmethodik – Wie wird getestet?“ einteilen (Vgl. Hoffmann, (2013), S. 158).

Die Einteilung in die Prüfebenen basiert auf der Programmstruktur der Software und auf der Entwicklungsphase. Darin werden die Tests in Unit-Tests, Integrationstests, Systemtests und Abnahmetests unterschieden (Vgl. Hoffmann, (2013), S. 159).

*Unit-Tests*

In einem Unit-Test, auch als Modultest oder Komponententest bezeichnet, wird eine atomare Programmeinheit getestet. Eine Unit kann aus einzelnen Funktionen, Klassen, oder Paketen und Bibliotheken bestehen (Vgl. Hoffmann, (2013), S. 159).

*Integrationstest*

Die nächsthöhere Abstraktionsstufe in der Klasse „Prüfebene“ bilden die Integrationstests. Darin werden größeren Software-Komponenten, die sich aus einzelnen Programmmodulen zusammensetzen, getestet. Ein Integrationstest soll sicherstellen, dass die zusammengesetzten Programmkomponenten ein funktionsfähiges System ergeben. In der Praxis werden Integrationstests als „Big-Bang-Integration“, „Strukturorientierte Integration“ oder „Funktionsorientierte Integration“ ausgeführt (Vgl. Hoffmann, (2013), S. 163).

*Systemtest*

Sobald alle Teilkomponenten eines Software-Systems erfolgreich integriert sind, kann mit den Systemtest begonnen werden. Dabei wird das gesamte System getestet und auf die Einhaltung der im Pflichtenheft spezifizierten Eigenschaften überprüft. Im Systemtest wird die Software funktional gemäß den Anforderungen geprüft. Im Gegensatz zum Unit- oder Integrationstest bleibt die Code-Struktur unberücksichtigt. Faktoren, wie unvorhergesehene Fehler, unklare Anforderungen, eingeschränkte Debug-Möglichkeiten und eingeschränkte Handlungsfähigkeit erschweren die Systemtest und die Planung und Einrichtung der der Testumgebung sind sehr aufwendig (Vgl. Hoffmann, (2013), S. 166 f.).

*Abnahmetest*

Auch der Abnahmetest verifiziert und validiert die Software bezüglich der Vorgaben und Spezifikationen aus dem Pflichtenheft. Im Gegensatz zum Systemtest wird der Abnahmetest unter der realen Einsatzumgebung des Kunden und mit authentischen Daten des Auftraggebers durchgeführt. Beim Abnahmetest gibt der Auftraggeber die Richtlinien des Tests vor oder er führt den Abnahmetest selbst durch (Vgl. Hoffmann, (2013), S. 168 f.).



(Quelle: Hoffmann, 2013, S. 159)

Abb. 9.2: Die vier Prüfebenen des Software-Tests

Hinsichtlich der Prüfkriterien werden die Software-Tests nach inhaltlichen Aspekten eines Testfalls wie folgt klassifiziert.

*Funktionaler Software-Test*

In funktionalen Tests wird die Datenverarbeitung geprüft. Zu diesen gehören Funktionstests, Trivialtests, Crashtest, Kompatibilitätstest und Zufallstests (Vgl. Hoffmann, (2013), S. 170 f.).

*Operationale Software-Tests*

Bei diesen Tests wird das Software-System unter dem Aspekt des operativen Einsatzes validiert und verifiziert. Ergonomietests überprüfen beispielsweise die Benutzbarkeit, Installationstests die reibungsfreie Inbetriebnahme und Sicherheitstests sollen Sicherheitslecks aufzeigen bzw. nachweisen, dass vertrauliche Daten geschützt verarbeitet und gespeichert werden (Vgl. Hoffmann, (2013), S. 172).

*Temporale Software-Tests*

Hierzu gehören Komplexitätstests, Laufzeittests, Lasttests, Stresstests (Vgl. Hoffmann, (2013), S. 172 f.).

Die in den Tests eingesetzten Methoden und Techniken sind das dritte Merkmal zur Klassifizierung der Software-Tests. Dabei werden die Software-Tests in drei Kategorien eingeteilt.

*Black-Box-Tests*

In den Black-Box-Tests wird ausschließlich das Ein- und Ausgabeverhalten der Software geprüft. Die Testfälle werden anhand der Anforderungs- und Schnittstellenbeschreibung konstruiert. Die innere Code-Struktur wird nicht berücksichtigt (Vgl. Hoffmann, (2013), S. 173 f.).

*White-Box-Tests*

Für White-Box-Tests werden die Testfälle aus der inneren Programmstruktur abgeleitet. Dabei wird in kontrollflussorientierten oder datenflussorientierten White-Box-Tests unterschieden, abhängig nach welchem Aspekt der Programm-Code geprüft werden soll (Vgl. Hoffmann, (2013), S. 174).

*Gray-Box-Tests*

Hierbei werden die Testfälle anhand der Anforderungs- und Schnittstellenbeschreibung und der inneren Programmstruktur konstruiert (Vgl. Hoffmann, (2013), S. 174).

Der schematische Ablauf von Tests besteht aus den Phasen:

*Testvorbereitung*

In der Testvorbereitung werden die Testfälle ausgewählt, die Testumgebung eingerichtet und Testdaten bereitgestellt (Vgl. Frühauf et al., (2007), S. 36 ff.). Die Spezifikation der Testfälle, die Reihenfolge, in der die Test auszuführen sind und das Vorgehen werden in der Testvorschrift dokumentiert. Durch die Vorgaben in der Testvorschrift werden die Aufwendungen reduziert und die Durchführung des Tests transparent. Das ermöglicht, den Testfall jederzeit mit denselben Einstellungen zu wiederholen und das dokumentierte Testergebnis nachzuvollziehen (Vgl. Frühauf et al., (2007), S. 36, 72).

*Testausführung*

Während dieser Phase führt der Tester die spezifizierten Testfälle nach den Testvorgaben der Testvorschrift aus und dokumentiert die Testergebnisse (Vgl. Frühauf et al., (2007), S. 36 ff.).

*Testauswertung*

Diese Phase dient der Identifizierung von Soll‑ / Ist‑Abweichungen. Diese werden als Fehler im Testprotokoll dokumentiert (Vgl. Frühauf et al., (2007), S. 36).



(Quelle: Frühauf et al., 2007, S. 37)

Abb. 9.3: Schematischer Testablauf

Nach der umfangreichen theoretischen Einführung in die Thematik wird folgend die Durchführung der Tests zur Software „Progress Monitor“ beschrieben.

Ein Bestandteil des Konzepts zum Testen der Software ist, dass bereits in der Entwicklung Unit-Test geschrieben und vor der Auslieferung automatisiert ausgeführt werden. Daher beschränkt sich die Beschreibung auf die Durchführung der Systemtests. Die Systemtest wurden als Black-Box-Tests, in denen das Ein- und Ausgabeverhalten der Software geprüft wird, ausgeführt.

## Testvorbereitung

In der Testvorbereitung wurden zunächst die Testfälle und die Durchführung der Tests in der Testvorschrift (Dokument „TSpec-Doc 1/001“ vom 25.07.2023) dokumentiert. Diese beschreibt die Spezifikation der Testfälle gemäß den Anforderungen aus der Anforderungsanalyse, die Reihenfolge, in der die Test auszuführen sind und die Anweisungen, zur Ausführung der Tests und zur Vorbereitung des Testgeschirrs. Durch die Testvorschrift ist erst die Durchführung der Tests in hoher Qualität möglich.

Die in der Testvorschrift beschriebenen Testfälle enthalten die Bezeichnung und den Zweck zum Testfall, den Anfangszustand, der Beschreibung der Testumgebung sowie die Eingaben, die Anweisungen zur Durchführung und die erwartete Ausgabe bzw. das erwartete Verhalten des Programms (Vgl. Frühauf et al., (2007), S. 38, 71).

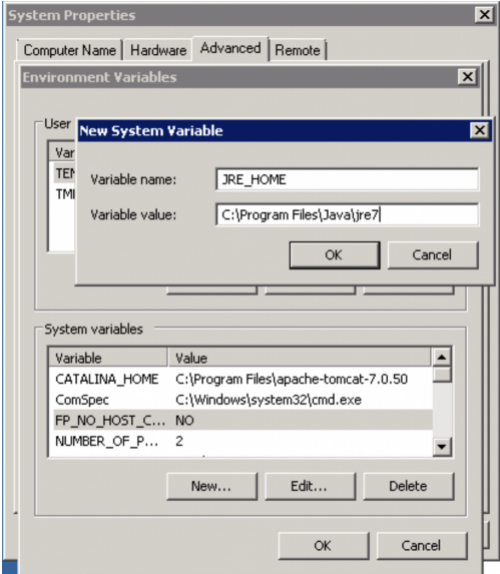
In der Testvorschrift ist jeder Testfall einer Testsequenz zugewiesen. Testsequenzen strukturieren die Testvorschrift, reduzieren den Aufwand in der Erstellung der Testvorschrift und dienen der zielorientierten und effizienten Durchführung der Tests. Eine Testsequenz besteht aus einem oder mehreren Testfällen und beschreibt einen Testabschnitt zu einer Softwarekomponente. In einer Testsequenz sind die Testfälle in funktioneller Reihenfolge angeordnet, so dass jeder Testfall die Voraussetzungen für den nachfolgenden erfüllt. Ferner können die Testsequenzen zweckorientiert definiert werden. Dabei entspricht die Reihenfolge der Testfälle in der Testsequenz der im Test notwendigen Reihenfolge. Die Anordnung der Testfälle wird so gewählt, dass die Ausführung eines Testfalls den folgenden Testfall vorbereitet oder das oder das Ergebnis des zuvor ausgeführten Testfall im folgenden Testfall anzeigt wird. Zu jeder Testsequenz sollten die Testfälle tabellarisch erfasst werden. Die Testfälle sind so zu beschreiben, dass die Beschreibung möglichst auch für das Testprotokoll verwendet werden kann (Vgl. Frühauf et al., (2007), S. 69 ff.).

Nach der Definition der Testfälle konnte die Testumgebung eingerichtet und die Software für die Tests bereitgestellt werden. Dazu wurden auf dem Testsystem die in der folgenden Tabelle genannten Programme installiert.

Tab. 9.1: Übersicht über die zu installierenden Programme

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Programm | Version | Zweck |
|  | Apache Tomcat | 7.0 | stellt die Dienste des Webservers für Java-Programme zur Verfügung |
|  | Java  inklusive JDK, JRE | 8.0 | Bereitstellen des JDK und der JRE |
|  | MySQL | Version 5.0.45-community-nt | RDBMS für die Software |
|  | Progress Monitor | Version: 12 Build: dev: 2023-07-06 Development version | Programm zur Überwachung und Kontrolle der Datenkonvertierung |

Die Einrichtung der Testumgebung und Bereitstellung der Software „Progress Monitor“ erforderten nach der Installation der dafür erforderlichen Software die Hinterlegung der für den Webserver Apache Tomcat benötigten Umgebungsvariablen im Betriebssystem und die Installation der Programmdateien.



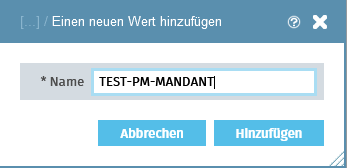
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.4: Einrichten der Umgebungsvariablen im Betriebssystem des Web-Servers

Für die verschlüsselte Kommunikation wurde im Anschluss das serverseitige SSL-Zertifikat installiert.

Mit der Installation der Datenbankobjekte und der Konfiguration der Software wurde die Einrichtung abgeschlossen (Vgl. Bayer, (2023), S. 1 ff.).

Nach der Installation des Web-Servers und der Bereitstellung der Software „Progress Monitor“, als Voraussetzung für die Testumgebung, waren im Programm die für die Tests erforderlichen Testmandanten und Benutzer anzulegen. Im „Progress Monitor“ werden die Zugriffrechte auf Mandanten und Programmkomponenten über Benutzerrollen und den mit diesen Benutzerrollen verknüpften Benutzerrechten gesteuert. Darüber hinaus können für jeden einzelnen Benutzer dedizierte Benutzerrechte vergeben werden. Der Verifikation der Benutzerverwaltung und dazugehörigen Benutzerrechten waren ein großer Teil der Testfälle zugewiesen.



(Quelle: Softzoll GmbH & Co. KG, 2023)

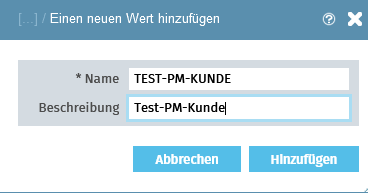
Abb. 9.5: Progress Monitor: Mandantenanalage im "Progress Monitor"



(Quelle: Softzoll GmbH & Co. KG, 2023)

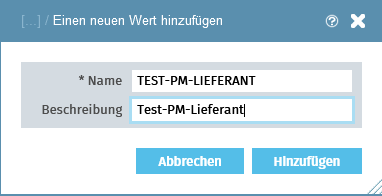
Abb. 9.6: Progress Monitor: Benutzeranalage im "Progress Monitor"

Anschließend waren die für die Tests erforderlichen Konten für die Lieferanten und Kunden sowie die dazu verwendeten Workflows anzulegen. Die Anlage dieser erfolgte nach den Vorgaben aus der Testvorschrift.



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.7: Progress Monitor: Anlage eines Testkunden



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.8: Progress Monitor: Anlage eines Testlieferanten



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.9: Progress Monitor: Anlage eines Workflows

Einen wesentlich Teil der Arbeit zur Testvorbereitung nimmt die Bereitstellung der für die Testfälle geeigneten Testdaten in Anspruch. Hier wurde für jeden Testfall in der Konvertierung eine Datei bereitgestellt. Dabei wurden Dateien mit anonymisierten Daten aus dem produktiv genutzten Daten-Konverter verwendet. Das Aufbereiten der Testdaten war zeitaufwendig, da für jeden Datentyp (entspricht einem Konvertertyp) eine Datei bereit zu stellen war.

Mit der Bereitstellung der Testdaten war die Phase der Testvorbereitung abgeschlossen und die Systemtest konnten durchgeführt werden.

## Testausführung

Die Systemtest wurden als Black-Box-Tests, die ausschließlich das Ein- und Ausgabeverhalten der Software prüfen, durchgeführt. Der Black-Box-Test ist funktionsorientiert, das bedeutet in ihm wird die Erfüllung der funktionalen Anforderungen nachfolgend genannten Kriterien geprüft:

*Funktionsüberdeckung*

Es wird jede Funktion in mindestens einem Testfall geprüft.

*Eingabeüberdeckung*

Jedes Eingabedatum ist in mindestens einem Testfall zu prüfen.

*Ausgabeüberdeckung*

Es ist jedes Ausgabedatum in mindestens einem Testfall auszugeben (Vgl. Frühauf et al., (2007), S. 45).



(Quelle: Frühauf et al., 2007, S. 37)

Abb. 9.10: Schematische Darstellung eines Black-Box-Tests

Die Testfälle orientierten sich dabei an der angestrebten Funktion bzw. der in der Anforderungsanalyse definierten Anforderung. Dabei entspricht jeder Testfall einer Anforderung und einer Testvorschrift.

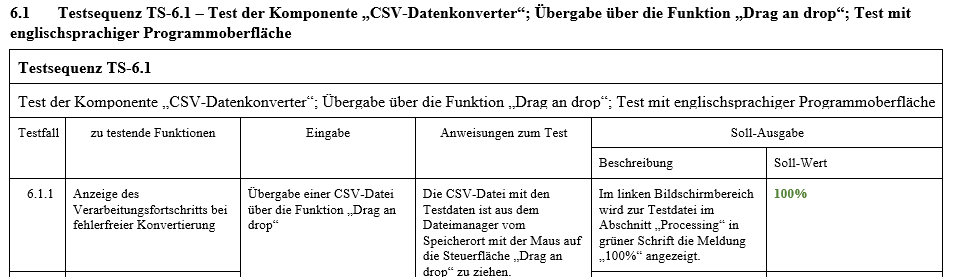
Exemplarisch werden die Tests der funktionalen Anforderungen zu der in der Testvorschrift spezifizierten Testsequenz

*TS 6.1 Test der Komponente „CSV-Datenkonverter“; Übergabe über die Funktion „Drag an drop“; Test mit englischsprachiger Programmoberfläche*

beschrieben.

In der Testvorschrift sind zu jeder Testsequenz der Zweck des Tests, die Referenzen zu den dazugehörigen Spezifikationen (Anforderungsdokumente, Installations- und Konfigurationsvorschriften, Programm-Dokumentationen etc.) die zu testenden Software-Komponenten, die erforderlichen Vorbereitungs- und Abschlussarbeiten und die Testfälle dokumentiert (Vgl. Frühauf et al., (2007), S. 69).

Die Testfälle sind in der Testsequenz tabellarisch erfasst. Darin ist jedem Testfall eine eindeutige Nummer zugewiesen. Für die Durchführung des Tests sind zu jedem Testfall die erwarteten Eingabedaten, die Testanweisung und die Soll-Ausgabe mit dem erwarteten Wert oder erwarteten Verhalten des Programms dokumentiert.



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.11: Abbildung zur Testsequenz TS-6.1 aus der Testvorschrift

Die Testsequenz TS-6.1 beinhaltet die drei Testfälle

*6.1.1 - Anzeige des Verarbeitungsfortschritts bei fehlerfreier Konvertierung*

In diesem Testfall sind durch den Tester die Anzeige des Verarbeitungsfortschritts, die Beschriftungen aller Bildschirm und die Beschriftungen der Bildschirmelemente und die Bildschirmausgaben bei einer fehlerfreien Konvertierung zu prüfen.

*6.1.2 - Anzeige des Verarbeitungsfortschritts bei fehlerhafter Konvertierung*

Die Testvorschrift zum Testfall ist mit der des Testfalls 6.1.1 identische, jedoch muss in diesem Testfall eine fehlerhafte Konvertierung getestet werden.

*6.1.3 - Anzeige der Detailinformationen zu einer Konvertierung*

Zum Testfall 6.1.3 sind die Detailinformationen einer Konvertierung, die fehlerfrei abgeschlossen wurde, zu prüfen.

Die Nummer eines jeden Testfalls setzt sich aus der Nummer der Testsequenz und einer fortlaufenden Nummer zusammen. Testfall 6.1.3 ist somit der 3. Testfall in der Testsequenz 6.1

Folgend wird die Durchführung des Testfalls 6.1.3 zur Testsequenz 6.1 beschrieben. Dieser Testfall wurde gemäß der Vorgaben aus der Testvorschrift mit dem Benutzer „Test-PM-Benutzer“ und mit der englischsprachigen Benutzeroberfläche (GUI) durchgeführt.

Im ersten Schritt wurde die zum Testfall 6.1.3 erstellte Testdatei an den CSV-Datenkonverter übergeben. Der Bezeichner der Testdatei ist nicht willkürlich gewählt, sondern in der Testvorschrift vorgegeben und setzt sich aus

dem Präfix „TF-“\_<Nummer des Testfalls>\_<Dateityp>\_<Testszenario>.<ENDUNG-DATEITYP> zusammen

Für den Testfall 6.1.3 und einer zu testenden CSV-Datei, die keine Fehler und die Daten zu mehreren Belegen enthält, ist die Dateibezeichnung somit:

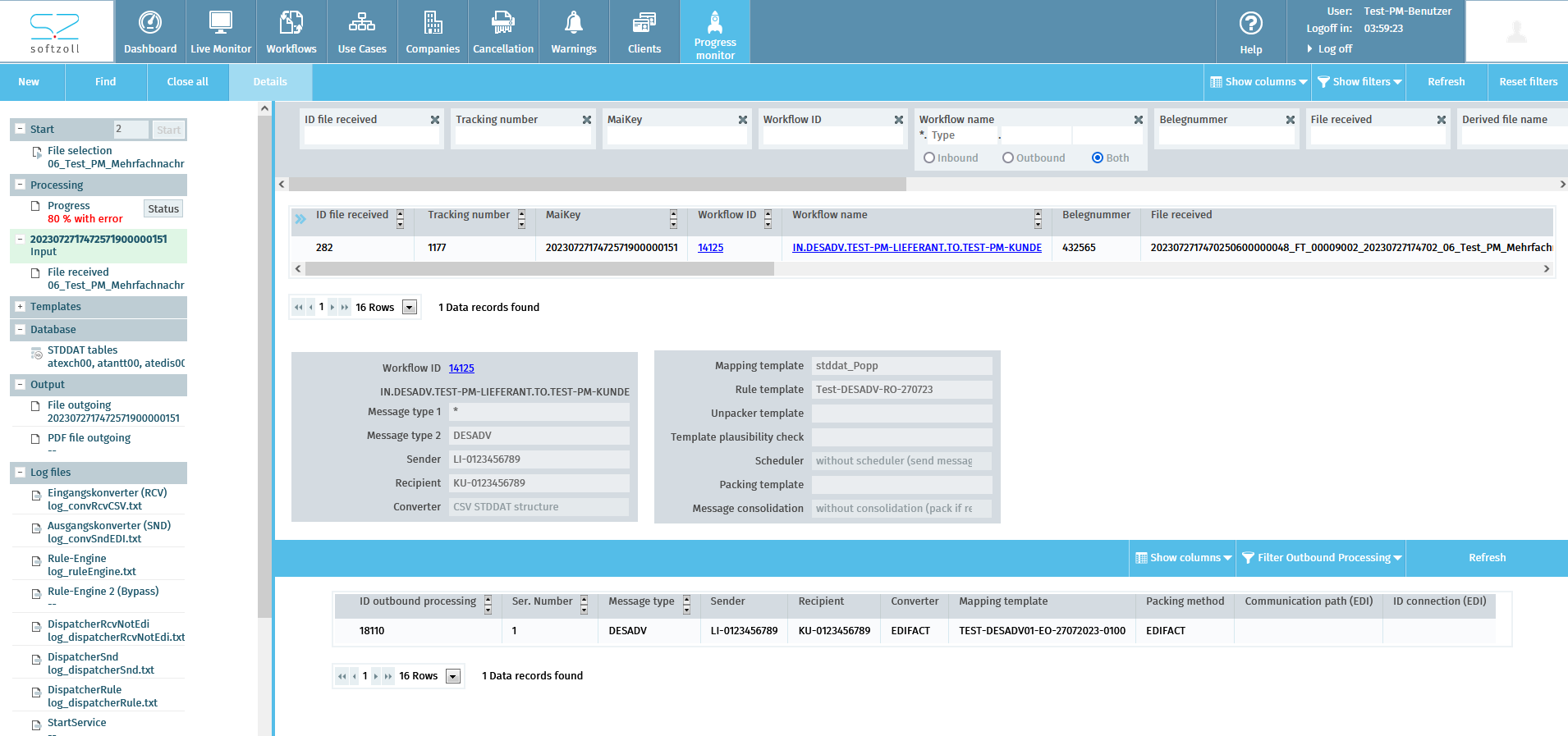
*TF-6.1.3\_CSV\_ohne\_Fehler\_Mehrfachnachricht.csv*



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.12: Übergabe der Test-Datei an den Datenkonverter, Testfall "TF-6.1.3"

Nach der Konvertierung dieser Datei wurden gemäß der zum Testfall 6.1.3 spezifizierten Vorgaben aus der Testvorschrift die Detailinformationen zur Konvertierung und die Beschriftung der Textfelder und Steuerelemente im Programm geprüft. Im Test konnte tatsächlich ein Fehler festgestellt werden. Die Tabellenspalte zur „Belegnummer“ ist in der englischsprachigen Benutzeroberfläche mit „Document number“ zu bezeichnen. Zu dieser Tabellenspalte wurde jedoch die deutsche Bezeichnung „Belegnummer“ angezeigt. Dieser Fehler ist gemäß der Testvorschrift nicht kritisch. Das bedeutet, dass anschließend weitere Tests ausgeführt werden konnten. Bei kritischen Fehlern, wie bei einem Ausnahmefehler (Exception) ist gemäß der Vorgaben aus der Testvorschrift der Test zu unterbrechen und der Fehler zu beheben.



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 9.13: Bildschirmausgabe zum Testfall 6.1.3

Der Test zum Testfall 6.1.3 bewies, wie wichtig Tests sind, auch wenn im Testergebnis lediglich ein nicht kritischer Fehler festgestellt wurde. Für die methodische und planmäßige Fehlerbeseitigung und für die Wiederholung des Test zu einem späteren Zeitpunkt sind der Test und dessen Ergebnis im Testbericht zu dokumentieren (Vgl. Frühauf et al., (2007), S. 72).

## Testauswertung

Jeder Test wird in einem Testbericht dokumentiert. Dieser enthält die Beschreibung der Testfälle und die Testresultate und basiert auf der Testvorschrift (Vgl. Frühauf et al., (2007), S. 34). Daher ist eine sorgfältige und detaillierte Testvorschrift für die Testläufe und für die Dokumentation von entscheidender Bedeutung.

Der Testbericht enthält unter anderem folgende Informationen:

*Testzusammenfassung*

Diese ist eine Übersicht zur verwendeten Testvorschrift, Sequenz-Nummer, Testfall-Nummer, zu den verwendeten Dokumenten und Namen der Teste sowie zur Projekt-Nummer und zum Projektnamen. Ferner enthält die Zusammenfassung die Empfehlung über den Umgang mit identifizierten Fehlern und die dazugehörige Begründung (Vgl. Frühauf et al., (2007), S. 71).

*Testprotokoll*

Im Testprotokoll sind die Durchführung des Tests, die Soll-Vorgaben, das Testergebnis und die Soll‑/Ist‑Abweichungen dokumentiert. Das Testprotokoll muss auch die Fehlermeldungen des Programms und die im Test identifizierten Fehler sowie die dazugehörigen Beschreibungen enthalten (Vgl. Frühauf et al., (2007), S. 71).

*Software-Komponenten*

Die Dokumentation der getesteten Software-Komponenten mit präziser Angabe zur Release‑ und Build‑Version ist für die Verständlichkeit und Transparenz des Tests essentiell und ermöglicht die Wiederholung des Testfalls sowie die Kontrolle zur Anwendung der richtigen Konfiguration (Vgl. Frühauf et al., (2007), S. 72).

Die genannten Inhalte des Testberichts beschränken sich auf die wesentlichen Informationen. Der zum Test erstellte Testbericht ist umfangreicher. Nach Durchführung des Tests wurden alle zum Test relevanten Informationen und das Testergebnis sowie die Soll‑/Ist‑Abweichungen im Testbericht dokumentiert. Der Anlage ist der Testbericht beigefügt[[3]](#footnote-3)

# Fazit

Am Ende möchte ich die Projektarbeit zusammenfassen, meine persönlichen Erfahrungen und Erkenntnisse, die ich in diesem Projekt sammeln konnte, aufzeigen und einen Ausblick auf die künftige Entwicklung geben.

Die Software „Progress Monitor“ wurde als unternehmensinternes Projekt termingerecht entwickelt. In dem für den Anwender freigegebenen Release sind die wichtigsten Anforderungen und Spezifikationen aus der Anforderungsanalyse und dem Pflichtenheft erfüllt. Durch die Anwendung des Dev-Ops-Konzepts einschließlich der der kontinuierlichen Integration („Continuous Integration“) und kontinuierlichen automatisierten Software-Auslieferung mittels der „Delivery Pipeline“ konnte die Zeit der Auslieferung funktionsfähiger Release Candidates (RC ) verkürzt werden. Der wesentliche Aspekt dieses Konzepts, die verbesserte Zusammenarbeit und Kommunikation zwischen der Softwareentwicklung und dem IT-Betrieb, war ein Garant für die Sicherstellung der Softwarequalität. Dadurch wurde die Anzahl der Fehler in der Software signifikant reduziert. Neben dem Projektmanagement, der Anforderungs- und Risikoanalyse sowie der Aufnahme der Ist-Situation gehörten auch die Mitarbeit am Entwurf und an der Spezifikation zu meinen Projektaufgaben. Ferner waren die Durchführung der Systemtests und Bereitstellung der Software ein wesentlicher Teil meiner Arbeit im Projekt.

Das Projekt ermöglichte mir, das im Studium erlernte Wissen aus den Bereichen Softwareentwicklung und Softwaretechnik sowie Betriebswirtschaftslehre und Projektmanagement zu verknüpfen und in der Praxis umzusetzen. Dabei konnte ich auch meinen Erfahrungsschatz aus anderen Projekten nutzen. Durch die praktische Anwendung des Wissens ergaben sich für mich neue Erkenntnisse und noch vorhandene Wissenslücken konnten geschlossen werden. Für mich ergab sich der größte Lerneffekt in der Anwendung des Dev-Ops-Konzepts. Im Studium konnte ich mir zu diesem Konzept bereits einiges Wissen aneignen und in einem Studienprojekt anwenden, jedoch erst in diesem Projekt erstmalig in der beruflichen Praxis nutzen. Dabei begeistern mich die positiven Effekte in der Zusammenarbeit und Erhöhung der Produktivität.

Die Software „Progress Monitor“ wird auch nach diesem ersten Projektabschnitt weiterentwickelt. Das ist darin begründet, dass die Daten-Konverter stetig weiterentwickelt werden bzw. neue Konverter das Portfolio erweitern.

An dieser Stelle möchte ich mich auch für die gute Projektbetreuung durch die Projektleitung und die gute Zusammenarbeit mit der Softwareentwicklung bedanken.

Anhang

Anforderungen aus dem Anforderungsdokument ReqSpec 1.001 vom 26.06.2023

Die wichtigsten Anforderungen an die Software „Progress Monitor“ sind in der folgenden Tabelle zusammengefasst.

Tab. 10.1: Anforderungsdokument ReqSpec 1.001 vom 26.06.2023

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Anforderung | Beschreibung | Kategorie |
|  | Anwendung soll unabhängig vom Betriebssystem nutzbar sein | Die Anwender arbeiten im Home-Office bzw. mobile Office und nutzen Endgeräte mit verschiedenen Betriebssystemen. | nicht  funktional |
|  | Die Konvertierungen sind in Echtzeit darzustellen | Die Anwender müssen die Datenkonvertierung und evtl. auftretende Fehler in Echtzeit verfolgen können. | nicht  funktional |
|  | Es sind die Konvertierungen zu allen verwendeten Dateikonvertern anzuzeigen. | Die Dateikonverter sind die Basis-Softwarekomponenten für die Datenkonvertierung. Derzeit werden Konverter für die Dateiformate Comma Separated Value (CSV), Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT), Fix Format (FF), Hypertext Markup Language (HTML), Intermediate Document (IDoc), Infor (Infor), JD Edwards (JDE), Tradacoms (TRC), ANSI X.12 (X.12) und Extensible Markup Language (XML) verwendet. In der Software sind die Konvertierungen zu allen genannten Dateiformaten anzuzeigen. | funktional |
|  | Sämtliche unterstützte Kommunikationsanbindungen sollen für den Anwender einzusehen sein. | Der Nachrichtenaustausch erfolgt über die Netzwerkprotokolle AS2, FTP, HTTP, HTTPS, SFTP, SMTP, X.400. In der Software soll der Benutzer alle genannten Protokolle einsehen können. | funktional |
|  | Der Benutzer muss die Inhalte der in der Datenbank verwendeten Arbeitstabellen einsehen können, | Die Daten werden in der Konvertierung in verschiedenen Tabellen einer SQL-Datenbank gespeichert. Diese Tabellen werden Arbeitstabellen genannt. Die Inhalte aus diesen Tabellen sind für den Anwender über das Frontend der Anwendung anzuzeigen. Dabei hat der Benutzer die Möglichkeit nach frei wählbaren Suchbegriffen in den Tabellen zu suchen und zu filtern. | funktional |

|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Anforderung | Beschreibung | Kategorie |
|  | Der Fortschritt der Konvertierung ist grafisch und als Prozentwert darzustellen. | Die Konvertierung erfolgt in mehreren Phasen nach der Methode Extract-Transform-Load (ETL). In der Software ist der Fortschritt zu jeder der Phasen als Balkendiagramm und als prozentualer Wert dazustellen. | funktional |
|  | Fehler in der Konvertierung sind anzuzeigen | Bei auftretenden Fehlern ist eine qualifizierte Fehlermeldung auszugeben. diese Fehlermeldung soll über die betreffende Software-Komponente und den Fehlertyp informieren. | funktional |
|  | Protokolle der Konverter und der für die Kommunikation verwendeten Softwarekomponenten sollen einsehbar sein. | Der Anwender soll zu Analysezwecken alle Protokolle der Konverter und Kommunikationsanwendungen einsehen können. | funktional |
|  | Die Konfiguration der für die Konvertierung verwendeten Workflows muss einzusehen sein und bearbeitet werden können. | Die Einstellungen zu jeder Konvertierung werden in Workflows vorgenommen. Für eine Analyse sollen die Workflowkonfigurationen einzusehen sein und durch den Benutzer bearbeitet werden können. | funktional |
|  | Der Benutzer soll mehrere zeitgleich ausgeführte Konvertierungen beobachten können. | In der täglichen Arbeit werden mehrere Konvertierungen zeitgleich ausgeführt. Diese sollen, gekapselt in einem dedizierten Prozess, parallel dargestellt werden.  Im Programm werden für jede Konvertierung werden die dazugehörigen Informationen separat visualisiert. | funktional |
|  | Mehrbenutzerfähigkeit | Die Anwendung soll von mehreren Benutzern parallel ausgeführt werden können. Dabei muss sich jeder Benutzer über ein eigenes Benutzerkonto am Programm anmelden können. | nicht funktional |
|  | sichere Benutzerauthentifizierung | Zur Wahrung des Datenschutzes und der Datensicherheit muss sich jeder Benutzer über einen Benutzernamen und Kennwort am Programm anmelden. | nicht funktional |
|  | verschlüsselte Kommunikation über HTTPS | Über HTTPS soll die verschlüsselte Kommunikation gewährleistet werden. | nicht funktional |
|  | autorisierte Zugriffe über dedizierte Benutzerrechte und -rollen | Die Benutzeraktivitäten sollen durch dedizierte Benutzerrechte und -rollen autorisiert werden | nicht funktional |
|  | detaillierte Protokollierung der Benutzeraktivitäten | In Protokolldateien (Logdateien) sollen die Benutzeraktivitäten zur Anmeldung am Programm zu Änderungen der Konfigurationen an den Workflows, Kommunikationsverbindungen und den Benutzerkonten und zur Durchführung der Konvertierungen protokolliert werden. Die Protokolldateien werden in TXT-Format und UTF-8 Zeichenkodierung erstellt. Jeder Eintrag in der Logdatei wird mit einem Datums-/Zeitstempel versehen. Jeden Tag um 00:00 Uhr wird die am Tag zuvor erstellte Datei archiviert und eine neue Protokolldatei erstellt. | nicht funktional |
|  | Anzeige des Verarbeitungsfortschritts bei fehlerfreier Konvertierung | in grüner Schrift: Anzeige der Meldung „100%“  Beispielwert:  „**100%**“ | nicht funktional |
|  | Anzeige des Verarbeitungsfortschritts bei fehlerhafter Konvertierung in der deutschsprachigen Programmoberfläche | in roter Schrift: Anzeige des Fortschritts zur Konvertierung als Prozentwert und die Meldung  {**Prozentwert**}„**mit Fehler**“  Beispielwert:  „**80% mit Fehler**“ | nicht funktional |
|  | Anzeige des Verarbeitungsfortschritts bei fehlerhafter Konvertierung in der englischsprachigen Programmoberfläche | in roter Schrift: Anzeige des Fortschritts zur Konvertierung als Prozentwert und die Meldung  {**Prozentwert**}„**with errors**“  Beispielwert:  „**80% with errors**“ | nicht funktional |
|  | Beschriftung der Textfelder und Steuerelemente im Programm | Beschriftung der Textfelder und Steuerelemente im Programm erfolgt gemäß Anforderungs-Dokument Referenz: ReqSpec 2.001 vom 12.04.2019 | nicht funktional |

Risikokatalog

Folgende Tabelle enthält einen Auszug aus dem Risikokatalog

Tab. 10.2: Risikokatalog

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Risiko | Wirkung | Ewk | Schadensausmaß | Maßnahme |
|  | Fehlerhaftes Projektmanagement | Nichterreichen der Projektziels  Terminüberschreitung | Hoch | Schwer: kein Projektabschluss  Überschreitung des Finanz‑ und Zeit‑Budgets | Effektives Projektmanagement |
|  | Unklarer Projektumfang |  | Hoch | Schwer: kein Projektabschluss  Überschreitung des Finanz‑ und Zeit‑Budgets  Nichterfüllen der Anforderungen | gut organisierter und strukturiertes Projektmanagement  Erstellen eines detaillierten Lastenhefts  Erstellen eines detaillierten Pflichtenhefts  sorgfältig ausgearbeiteter Projektantrag und der Projektauftrag |
|  | Fehlende Personalressourcen | Terminüberschreitung | Hoch | Mittel: Überschreitung des Finanz‑ und Zeit‑Budgets | Pufferzeiten in den Arbeitspaketen  Ermitteln der Abhängigkeiten zu den Aufgaben  Planung der Substitution von Mitarbeitenden |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Risiko | Wirkung | Ewk | Schadensausmaß | Maßnahme |
|  | Fehlende Festlegung der Funktionen und Rollen | Verzögerungen im Projektablauf  Terminüberschreitung  Nichterreichen der Projektziels  Überschreitung des finanziellen Budgets | Hoch | Schwer: kein Projektabschluss  Überschreitung des Finanz‑ und Zeit‑Budgets | präzise Bestimmung der Funktionen und Rollen und Zuweisung dieser an die Ausführenden |
|  | Fehler in der Anforderungserhebung | Annahme falscher Anforderungen; Anforderungen werden nicht erfüllt | Hoch | Schwer: Anforderungen des Kunden werden nicht erfüllt | Bestimmen der kompetenten Ansprechpartner und Entscheidungsträger  Gründliche Vorbereitung auf die Interviews  Führen und Dokumentieren der Interviews  Pflichtenheft  Entwickeln von Mockups und Prototypen  Intensive Kommunikation mit dem Kunden und den Shareholdern |
|  | Änderung der Anforderungen | Nichterreichen der Projektziels  Terminüberschreitung  Verzögerungen im Projektablauf  fehlende bzw. unzureichende Funktionalitäten in der Software | Hoch | Mittel: unzureichende/fehlerhafte Software  Nachbesserung erforderlich | sorgfältige Durchführung des Requirements Engineering und Change Managements |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr. | Risiko | Wirkung | Ewk | Schadensausmaß | Maßnahme |
|  | Fehler in der Implementierung | Fehler in der Ausführung der Software; fehlerhafte Software-Funktionalitäten  geforderte Funktionalitäten werden nicht oder fehlerhaft umgesetzt | Hoch | Schwerkalkulierbar  Kosten = Personalstunden \* Stundensatz | Unit-Test,  Integrationstests,  Systemprüfungs-Test,  Abnahmeprüfung  Dokumentation der Tests |
|  | Fehlende bzw. mangelhafte Durchführung der Qualitätssicherung | mangelhafte bzw. fehlerhafte Software  fehlende Funktionalitäten in der Software | Mittel: | Mittel: fehlerhafte Software  Nachbesserung erforderlich | Planung, Konzeption und Definition der Qualitätssicherung  Einbeziehung des Kunden in die Qualitätssicherung  agile Softwareentwicklung mit kontinuierlichen Tests und frühzeitiger Entwicklung von Prototypen |
|  | Fehlendes Bewusstsein für die Produktqualität | mangelhafte bzw. fehlerhafte Software  fehlende Funktionalitäten in der Software | Mittel | Mittel: fehlerhafte Software  Nachbesserung erforderlich | Motivation der Mitarbeitenden zu qualitätsorientiertem Handeln |

Testbericht zum Test der Software „Progress-Monitor“

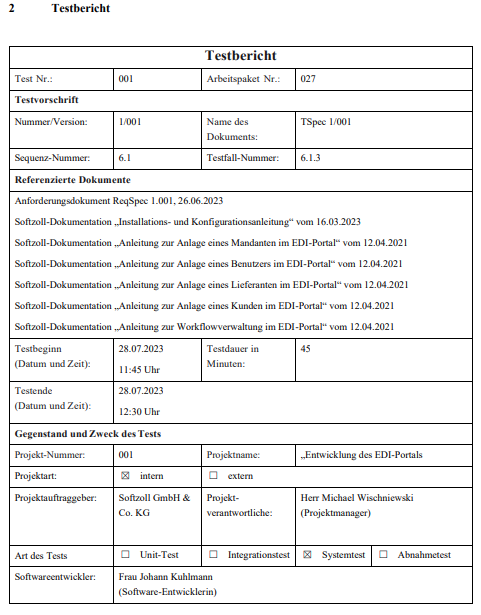
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.1: Testbericht zum Test der Software „Progress Monitor“, Seite 2

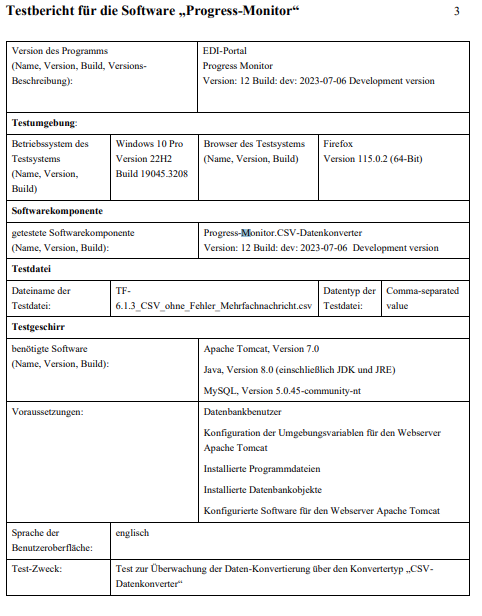
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.2: Testbericht zum Test der Software „Progress Monitor“, Seite 3

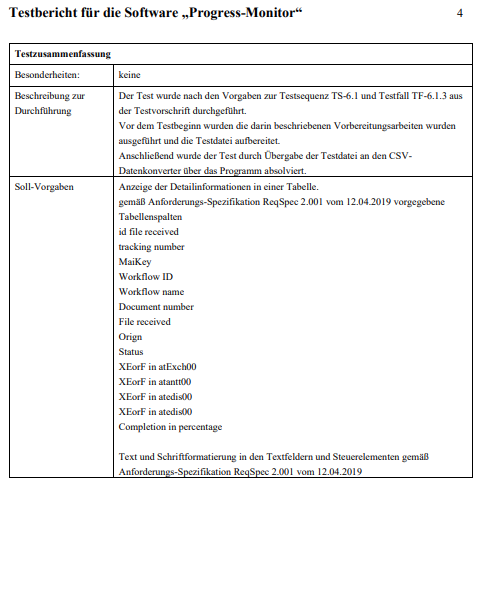
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.3: Testbericht zum Test der Software „Progress Monitor“, Seite 4

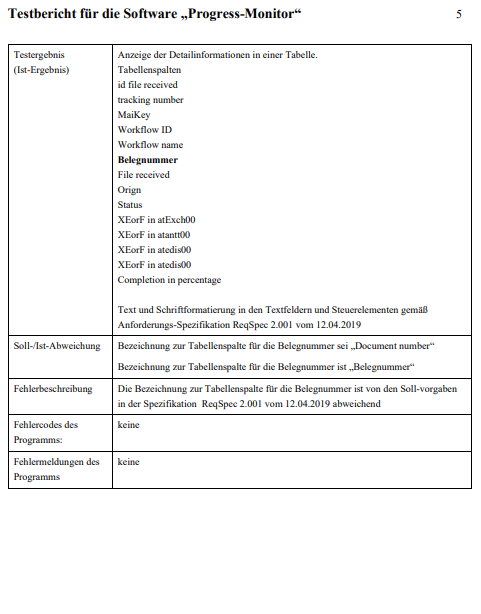
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.4: Testbericht zum Test der Software „Progress Monitor“, Seite 5

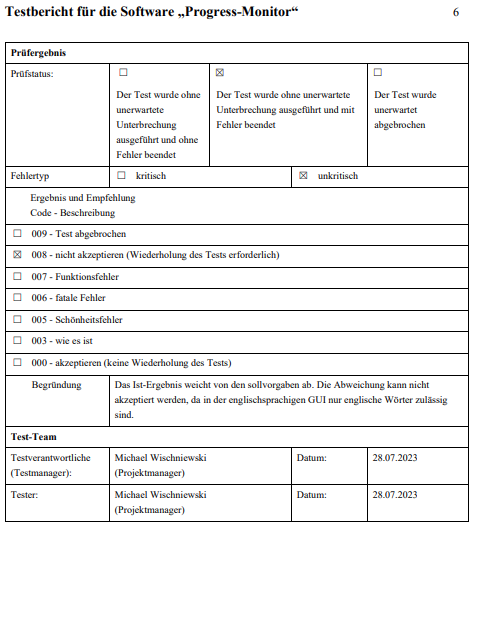
(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.5: Testbericht zum Test der Software „Progress Monitor“, Seite 6

BPMN zum Prozess „Softwaretest“



(Quelle: Softzoll GmbH & Co. KG, 2023)

Abb. 10.6: BPMN zum Prozess „Softwaretest“

Literaturverzeichnis

Aichele, C., & Schönberger, M. (Hrsg.). (2014). *App4U: Mehrwerte durch Apps im B2B und B2C*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-8348-2436-3

Alpar, P., Alt, R., Bensberg, F., & Czarnecki, C. (2023). *Anwendungsorientierte Wirtschaftsinformatik: Strategische Planung, Entwicklung und Nutzung von Informationssystemen*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-40352-2

Alt, R., Auth, G., & Kögler, C. (2017). *Innovationsorientiertes IT-Management mit DevOps*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-18704-0

Bayer, M. (2023). *Installation von Tomcat und EDI-Portal*. Softzoll GmbH & Co. KG.

Bechmann, R., & Landerer, S. (2010). *Qualitätsmanagement und kontinuierlicher Verbesserungsprozess*. Bund-Verl.

Beifuss, A., & Holzbaur, U. (2020). *Projektmanagement für Studierende: Strategie und Methode für ein erfolgreiches Studium*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-32664-7

Broy, M. (2023). *Logische und Methodische Grundlagen der Entwicklung verteilter Systeme: Unter Mitarbeit von Alexander Malkis*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-67317-1

Burnus, H. (2008). *Datenbankentwicklung in IT-Berufen: Eine praktisch orientierte Einführung mit MS Access und MySQL ; [mit Online-Service zum Buch]* (1. Aufl). Vieweg.

Carle, G., Günther, S., Herold, N., & Posselt, S. (2013). *Was\_ist\_Subversion\_TU\_Munich.pdf*. Technische Universität München. https://www.net.in.tum.de/pub/grnvs/2013/svnintro.pdf

Frick, D., Gadatsch, A., & Schäffer-Külz, U. G. (2008). *Grundkurs SAP ERP*. Vieweg+Teubner. https://doi.org/10.1007/978-3-8348-9264-5

Frühauf, K., Ludewig, J., & Sandmayr, H. (2007). *Software-Prüfung: Eine Anleitung zum Test und zur Inspektion* (6.). vdf Hochschulverlag AG.

Halstenberg, J., Pfitzinger, B., & Jestädt, T. (2020). *DevOps: Ein Überblick*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-31405-7

Herrmann, A. (2022). *Grundlagen der Anforderungsanalyse: Standardkonformes Requirements Engineering*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-35460-2

Hoffmann, D. W. (2013). *Software-Qualität*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-35700-8

Kaufmann, J., & Mülder, W. (2023). *Grundkurs Wirtschaftsinformatik: Eine kompakte und praxisorientierte Einführung*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-37937-7

Kirner, E., Armbruster, H., & Kinkel, S. (2006). *Kontinuierlicher Verbesserungsprozess-Baustein zur Prozessinnovation in KMU: Nutzung und Effekte von KVP im Verarbeitenden Gewerbe*. Mitteilungen aus der ISI-Erhebung-Modernisierung der Produktion.

Krallmann, H., Schönherr, M., & Trier, M. (2013). *Systemanalyse im Unternehmen* (6. Aufl.). Oldenbourg Verlag München Wien.

Kusay-Merkle, U. (2018). *Agiles Projektmanagement im Berufsalltag: Für mittlere und kleine Projekte*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-56800-2

Meier, A. (2010). *Relationale und postrelationale Datenbanken*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-05256-9

ORACLE. (2023). *ORACLE Java Tutorials*. https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html. abgerufen am 23.07.2023 um 12:56 Uhr

Pekša, J. (2018). Extensible Portfolio of Forecasting Methods for ERP Systems: Integration Approach. *Information Technology and Management Science*, *21*, 64–68. https://doi.org/10.7250/itms-2018-0010

Scheer, A.-W., Abolhassan, F., Jost, W., & Kirchmer, M. (Hrsg.). (2003). *Change Management im Unternehmen*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-19020-9

Simons, A., Niehaves, B., Niehaves, B., Reimer, K., & vom Brocke, J. (2009). RECONSTRUCTING THE GIANT: ON THE IMPORTANCE OF RIGOUR IN DOCUMENTING THE LITERATURE SEARCH PROCESS. *AIS Electronic Library*. https://aisel.aisnet.org/ecis2009

SOPHIST GmbH, & Rupp, C. (2013). *Systemanalyse kompakt*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-35446-5

Timinger, H. (2015). *Wiley-Schnellkurs Projektmanagement*. Wiley-VCH Verlag GmbH & Co. KGaA.

Valentini, U., Weißbach, R., Fahney, R., Gartung, T., Glunde, J., Herrmann, A., Hoffmann, A., & Knauss, E. (2013). *Requirements Engineering und Projektmanagement* (A. Herrmann, E. Knauss, & R. Weißbach, Hrsg.). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29432-7

Wack, J. (2007). *Risikomanagement für IT-Projekte* (1. Aufl). Dt. Univ.-Verl.

Abschließende Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig, ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedan­ken sind als solche kenntlich gemacht.

Berlin, den 25. August 2023

1. Verweis zu den Anforderungen aus dem Anforderungsdokument ReqSpec 1.001 vom 26.06.2023. [↑](#footnote-ref-1)
2. Verweis zum Risikokatalog. [↑](#footnote-ref-2)
3. Verweis auf den Testbericht zum Test der Software „Progress-Monitor“ [↑](#footnote-ref-3)